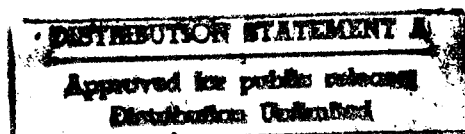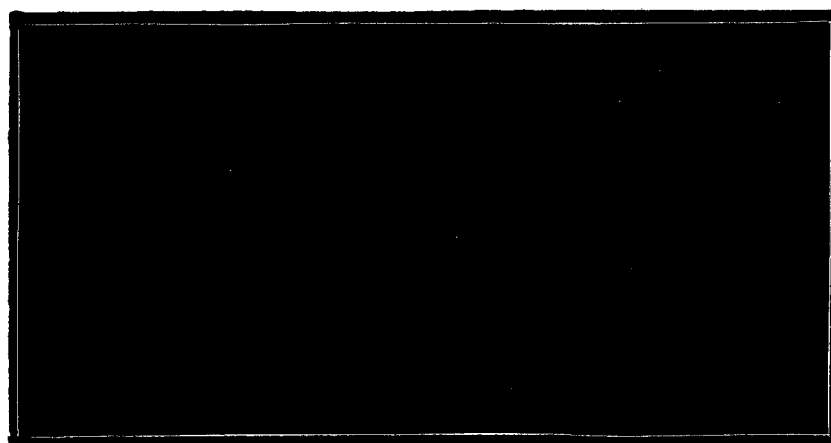DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOA/ENS/95M-04

PATIENT/AIRCRAFT FORECASTING
FOR THE STRATEGIC AEROMEDICAL EVACUATION
"LIFT-BED" PROCESS

THESIS

Donald F. Kimminau, Captain, USAF

AFIT/GOA/ENS/95M-04

Approved for public release; distribution unlimited

19950503 082

REPORT DOCUMENTATION PAGE

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE March 95	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE
PATIENT/AIRCRAFT FORECASTING FOR THE STRATEGIC AEROMEDICAL EVACUATION "LIFT-BED PROCESS

5. FUNDING NUMBERS

6. AUTHOR(S)

Donald F. Kimminau, Capt, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology,
WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GOA/ENS/95M-04

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

US TRANSCOM/GPMRC
505 D St., Scott AFB, IL 62225

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The goal of this research is to develop a tool for the Global Patient Movement Requirements Center (GPMRC) to efficiently forecast Aeromedical Evacuation (AE) aircraft and schedule patients in the *lift-bed* process. During a contingency, GPMRC is responsible for requesting required airlift, and providing *lift-bed* candidates to the Theater Patient Movement Requirements Center. The objective is to evacuate all waiting patients while minimizing the number of C-141 aircraft which must to be dedicated to the AE mission. In the solution process, a short-term forecast of patient movement requirements is input, including number of patients, bed and aircraft availability, and hub preferences. A mixed integer linear program is formulated, and solved. Upon reaching optimality or a user-specified node limit, the user analyzes the solution and determines if the search will be continued. At process completion, a solution file is produced which contains a feasible schedule which minimizes the number of dedicated C-141s required. The programs developed to assist the solution process were tested with 16 data sets. While optimal solutions are not guaranteed to be found for every case, the process quickly produces good, feasible solutions and provides timely, valuable information to GPMRC so it can request airlift and provide lift-bed candidates to appropriate agencies.

14. SUBJECT TERMS

Integer Programming, Aeromedical Evacuation

15. NUMBER OF PAGES
86

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to *stay within the lines* to meet *optical scanning requirements*.

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	-	Contract	**PR**	- Project
G	-	Grant	**TA**	- Task
PE	-	Program Element	**WU**	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

DOD	-	See DoDD 5230.24, "Distribution Statements on Technical Documents."
DOE	-	See authorities.
NASA	-	See Handbook NHB 2200.2.
NTIS	-	Leave blank.

Block 12b. Distribution Code.

DOD	-	Leave blank.
DOE	-	Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
NASA	-	Leave blank.
NTIS	-	Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

AFIT/GOA/ENS/95M-04

PATIENT/AIRCRAFT FORECASTING FOR THE STRATEGIC AEROMEDICAL EVACUATION "LIFT-BED" PROCESS

THESIS

Presented to the Faculty of the Graduate School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Operations Research

Donald F. Kimminau, B.S.

Captain, USAF

March, 1995

Approved for public release; distribution unlimited

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

THESIS APPROVAL

STUDENT: Donald F. Kimminau   CLASS: GOA 95M

THESIS TITLE: Patient/Aircraft Forecasting for the Strategic Aeromedical Evacuation
"Lift-Bed" Process.

DEFENSE DATE: 23 February 1995
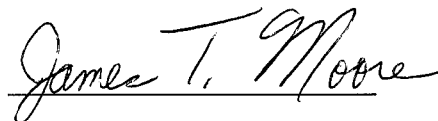
COMMITTEE:	NAME/DEPARTMENT	SIGNATURE
Advisor	LtCol James T. Moore/ENS	
Reader	Maj Lee J. Lehmkuhl/ENS	

Acknowledgments

I am extremely grateful to a number of people for assisting and supporting me in my thesis effort. All the personnel at the Global Patient Movement Requirements Center, especially LtCol Phil Mahlum, were very helpful in proving data and other information to ensure this effort benefits the Aeromedical Evacuation System as much as possible.

Additionally, I would like to express my appreciation and admiration for my advisor, LtCol Jim Moore. LtCol Moore furnished not only academic expertise for my thesis and coursework, but, more importantly, he provided motivation during my entire experience at AFIT. I would also like to thank Major Lee Lehmkuhl, my reader, for his assistance in providing input to improve the writing and research overall.

Most importantly, I wish to praise my family for their unending support and confidence. Without the extra work and encouragement supplied by my wife during the arduous time here at AFIT, I would not have completed my work here as well as I have. Thank you, Teresa. Also pushing me was the continual pride shown by my wonderful boys, Jared and Joseph. Their love and appreciation was apparent during play time, work time and especially prayer time. Thanks guys!

Donald F. Kimminau

Table of Contents

List of Tables

List of Figures

AFIT/GOA/ENS/95M-04

Abstract

The goal of this research is to develop a tool for the Global Patient Movement Requirements Center (GPMRC) to efficiently forecast Aeromedical Evacuation (AE) aircraft and schedule patients in the *lift-bed* process. During a contingency, GPMRC is responsible for requesting required airlift, and providing *lift-bed* candidates to the Theater Patient Movement Requirements Center. The objective is to evacuate all waiting patients while minimizing the number of C-141 aircraft which must to be dedicated to the AE mission.

In the solution process developed, the user inputs a short-term forecast of patient movement requirements, including number of patients, bed and aircraft availability, and hub preferences. A mixed integer linear program is formulated, and solved. Upon reaching optimality or a user-specified node limit, the user analyzes the solution and determines if the search will be continued. At process completion, a solution file is produced which contains a feasible schedule which minimizes the number of dedicated C-141s required.

The programs developed to assist the solution process were tested with 16 data sets. While optimal solutions are not guaranteed to be found for every case, the process quickly produces good, feasible solutions and provides timely, valuable information to GPMRC so it can request airlift and provide lift-bed candidates to appropriate agencies.

PATIENT/AIRCRAFT FORECASTING FOR THE STRATEGIC AEROMEDICAL EVACUATION "LIFT-BED" PROCESS

I. Introduction

An unfortunate consequence of any military contingency is the infliction of injuries to personnel. A vital determinant of the United States' warfighting ability has been and will be our ability to treat injured combatants. By quickly returning able soldiers to battle, we can keep a more experienced force in the field. Also, efficiently bringing home soldiers who cannot return to their duties can greatly enhance the morale of all troops. In addition, our prompt treatment of all injured combatants enhances the credibility of, and increases national support for, our military efforts.

Since 1949, it has been Department of Defense (DOD) policy to evacuate casualties by air when it is available and the patients injuries do not prevent air transport (16). In 1993, DOD tasked US Transportation Command (USTRANSCOM) to consolidate the control of medical regulation and aeromedical evacuation (AE) of patients of the US armed forces (4:3). The assets, including aircraft, aircrew and aeromedical crews, for the movement of injured soldiers from the combat zone Area of Responsibility (AOR) back to the Continental United States (CONUS) will be provided by USAF Air Mobility Command (AMC). AMC will:

serve as the force provider for all non-theater assigned active duty and Air Reserve Component (ARC) AE resources supporting intertheater interface or intertheater AE (5:Q-10).

Fortunately, during our most recent conflicts, the AMC AE system has not been fully exercised. Nevertheless, much emphasis has been placed on ensuring this system will be fully capable of handling patient transportation requirements during future conflicts. Most of the recent studies regarding the AE system have evaluated the number and type of aircraft required to support the mission. One recommendation has been to integrate evaluation techniques into a single computer operating environment (10:68). This will allow quicker solutions and real-time scheduling and analysis. Additionally, due to changes in the structure and operating policy of the AE system, elements of USTRANSCOM now need the ability to forecast and schedule aircraft and patients in the "lift-bed" process.

Background

In a contingency, injured combatants are placed into the in-theater medical treatment system. In 1979, US Air Forces Europe developed this system as a four echelon combat care concept patterned after a similar North Atlantic Treaty Organization system (7:10). The main objective of the medical system is to treat patients as close to the front as possible, and then return them to battle or evacuate them as conditions permit. The first echelon, 1E, consists of immediate first aid applied by the injured combatant (self-aid) or fellow soldiers (buddy-care). If required, patients are then taken to an appropriate Casualty Collection Point or a second echelon (2E) medical treatment facility. The 2E facilities have limited capability so, after patients have been stabilized, it may be necessary

to transport them to a hospital more able to treat their injuries. If so, the patients are moved to third echelon (3E) hospitals which are better equipped and further removed from the battlefield than the lower echelon facilities; however, they are still within the combat zone. The combat zone is defined as the "area required by combat forces for the conduct of operations" (6:2). The responsibility for transportation of the casualties in the combat zone rests with their parent service. Further casualty movement is the duty of the AMC AE system. This includes evacuating the patients from the combat zone to the fourth echelon (4E) hospitals located in the communications zone (COMMZ) and, if necessary, to CONUS hospitals which are referred to as 5E facilities in the four echelon medical care system (7:10-37). The COMMZ is the most rearward area of the theater of operations. It "contains lines of communications, establishments for supply, evacuation and other agencies required for the immediate support and maintenance of field forces" (6:2).

If at any time during treatment at the appropriate medical facilities, it is determined that the casualty, after treatment, will not be able to return to combat within a specified time limit, arrangements will be made for that patient to be evacuated back to the CONUS. The time limit is established by the theater commander and is primarily determined by the number of beds available in the theater (5:Q-8). This theater evacuation policy is a key element in defining AE requirements.

The assignment of patients to hospitals is performed by the medical regulation system. In the combat zone, this task is the responsibility of the theater Joint Medical Regulating Office (JRMO). The in-theater AE Coordination Center (AECC) manages the

AE mission and aircraft operations. Since the 1993 DOD initiative, theater commanders have the ability to integrate the regulating duties of the JRMO and the AECC intratheater aircraft operations into a Theater Patient Movement Requirements Center (TPMRC). When established, the TPMRC will acquire *theater* beds and airlift and combine them into "lift-beds" (5:Q-20). This "lift-bed" concept was instituted to evaluate all available lift and bed assets and to ensure that patients are evacuated in the most efficient manner to locations equipped and able to handle the patients' specific injuries (2:2).

Regulating duties for movements between 3E and 4E, and between 4E and CONUS facilities are accomplished jointly by the JRMO, or TPMRC, and the Global Patient Movement Requirements Center (GPMRC), located at Scott AFB, Illinois. USTRANSCOM established the GPMRC to combine the medical duties of the Armed Service Medical Regulating Office (ASMRO) and the global AE airlift operations into a single office. GPRMC will acquire *global* beds and AE airlift and, like the TPMRC, transform them into "lift-beds" (5:Q-19). Once it has been established that the patient must be evacuated to the United States, GPMRC, using the "lift-bed" concept, determines an appropriate CONUS medical facility and the required airlift to transport the patient to that facility.

Patients and hospital beds are categorized by injury type. These categories are Medicine, Psychiatry, Surgery, Neurosurgery, Maxillofacial, Opthamology, Thoracic Surgery, Urology, Orthopedics, Spinal Cord Injury, Burns, Pediatrics, and OB/GYN (12:2). The available beds of designated CONUS hospitals which are part of the National Disaster Medical System (NDMS) are also categorized in this manner. The NDMS was

established "to augment the nation's emergency medical response capability" (7:39). It consists of around 1000 US military, Veteran Administration, and civilian hospitals and is capable of handling a total of more than 100,000 patients.

The "lift-bed" candidates designated by GPMRC are relayed back to the theater where the JRMO, or TPMRC, selects which of the resources to assign to the individual patients (5:Q-20). They then authorize movement of the patients when they are considered able to withstand the flight. From 3E hospitals, patients are either routed directly to an aeromedical staging facility (ASF) or to 4E facilities to be further treated and stabilized. Once stabilized at the 4E facility, they are routed to the ASF. At the ASF, the patients will wait for an available aircraft to transport them to the specified CONUS hospitals.

Along with scheduling patients and aircraft, the in-theater TPMRC is responsible for monitoring patient movement. From the various ASFs, the casualties are taken to the nearby airfields and loaded onto the appropriate aircraft. The current policy is to evacuate all patients when they are released to travel by the medical regulating system. Initially, USAF C-141 Starlifters will be the primary aircraft designated to fly the AE missions. AMC will attempt to fill all AE requirements with *retrograde* C-141 missions. These are aircraft scheduled to return to the CONUS after transporting combat materiel and personnel to the AOR. If the AE requirements cannot be filled by the retrograde C-141s, *dedicated* C-141s will be scheduled to complete the tasking (11). The dedicated C-141s are devoted to AE for the entire mission, and will only be scheduled to fly to the AOR airports and return to the CONUS.

Recent studies have shown shortfalls in the ability of USAF military cargo aircraft to accomplish both the AE mission and their other wartime responsibilities. Because of this, AMC's future AE plans are based upon the ability to use the Civil Reserve Air Fleet (CRAF). If the CRAF is activated by the Secretary of Defense, AMC plans to use 20 Boeing 767s as the primary aircraft of the AE system (11). These aircraft will be reconfigured civilian airliners and can hold approximately 100 litter patients, each. While the civilian companies retain operational control over their resources, AMC will be responsible for mission allocation and scheduling (6:25).

The AE aircraft will depart the COMMZ and travel to CONUS airports. The CONUS AE system has a hub and spoke design. Destinations of the incoming overseas aeromedical aircraft will be hub airports. Upon debarkation at these airports, the patients will be ground transported to nearby hospitals or will again be air transported to the spoke airfields which are closest to the various other NDMS facilities. This system allows for shorter total travel time for the majority of the patients.

Problem Statement

GPMRC currently does not have an effective means to forecast and schedule patients and aircraft for evacuation from the in-theater ASFs to the CONUS hub airports. Given a forecast set of patient transportation requirements, the problem is to efficiently evacuate the casualties while minimizing the number of dedicated C-141 aircraft required to fulfill those requirements. The major constraints facing the scheduler are the limited number of available aircraft and CONUS hospital beds. This research develops a process that allows GPMRC personnel to input the various AE resources, requirements and

constraints and to produce the forecasts, for the TPMRC, of the appropriate "lift-bed" possibilities. The "lift-beds" can then be used by the TPMRC to schedule the patients and aircraft to accomplish the aeromedical evacuation to the CONUS.

Scope

This research models and analyzes only the COMMZ to CONUS portion of the AE mission. It does not include intratheater evacuation in the combat zone or the patient redistribution process in the CONUS. Using the current AMC AE plan, the model considers the CRAF aircraft, the retrograde C-141s, the dedicated C-141s, and the current capacities and capabilities of each aircraft type. This research also streamlines the work done previously in this area with primary attention given to the research done by Major Mike Loftus (10). In his research, Major Loftus describes a model with two parts: patient scheduling and aircraft assignment. The heuristic algorithm he developed iteratively solved the parts separately. Initially, patients are assigned aircraft on a first come-first serve basis. Next, the aircraft to airport assignment problem is solved and the solutions are inserted into a network model of the patient to aircraft assignment problem. He continued this iterative process until the user-defined termination criteria were met. An objective of the current research is to integrate patient scheduling and aircraft assignment and solve the resulting mathematical model. Within a reasonable amount of time, a schedule is generated which minimizes the number of dedicated C-141 missions and complies with resource constraints. Additionally Major Loftus considered only a subset of possible COMMZ and hub airports. The current research considers the entire AE CONUS hub airport system.

Assumptions

Aircraft. Realistically, an individual aircraft could fly between AOR airports to pick up patients or drop patients off at a combination of hub airports; however, this research assumes that an aircraft will be scheduled to fly only one route. That is, aircraft will transport patients from a single AOR airport to a single CONUS hub airport. Additionally, it is assumed that the user desires to use all available CRAF and C-141 retrograde aircraft before using any of the dedicated C-141s. All aircraft of a certain type are treated identically to other aircraft of that type. The assumed patient capacities of the aircraft are listed in Table 1.1.

Table 1.1 Aircraft Capacities

Aircraft Type	Capacity
CRAF Boeing 767s	100
Retrograde C-141s	60
Dedicated C-141s	90

Patients and Beds. All patients of a specific injury class at an AOR airport are treated identically. Beds assigned to injury classes at hub airports are also treated the same without regard to specific hospital location. This research does not model evacuation of patients with immediate priority transportation needs. AE missions will be flown to support the higher priority patients regardless of other movement plans.

Airports. Except for bed availability and user-defined preferences, hub airports are treated equally. When hub airports are preferred, airports are no longer treated equally, but split into preferred and non-preferred groups. Penalties are then assigned only

to the non-preferred hub airports. Also, no priority or penalty is associated with evacuating patients from any of the individual AOR airports.

Overview

The remaining chapters summarize the research venture. Chapter 2 provides a digest of other research efforts that relate to this research. Chapter 3 presents the mathematical programming model developed in this research. Next, Chapter 4 contains a summary of the results and analysis of the runs made using test data sets. The final chapter, Chapter 5, presents conclusions and recommendations for further research.

II. Literature Review

The purpose of this chapter is to summarize recent advances in the mathematical techniques used to evaluate AMC's wartime aeromedical evacuation (AE) system and the system's ability to handle the recently created forecasting and scheduling needs of GPMRC. GPMRC, along with AMC, has primary responsibility for the safe and efficient evacuation of U.S. wartime casualties from the theater of operations to the Continental United States (CONUS). Fortunately, the AE system has not been fully utilized during recent conflicts like Desert Storm. Nevertheless, AMC has placed much emphasis on ensuring the effectiveness of the AE process in possible future conflicts. If managed properly, the AE system can directly impact the nation's warfighting ability by keeping more experienced troops in the field, increasing the morale of the troops, and sustaining support from the entire nation.

Many recent research efforts have helped decision makers answer macro-level policy questions like type and number of aircraft needed to meet AE mission requirements. From these studies, AMC has obtained the information needed to operate the AE system adequately. To make it function as well as possible, however, there needs to be a system accessible to controlling agencies which facilitates real-time patient and aircraft scheduling. To date, there is no adequate scheduling system available.

This chapter concentrates primarily on information regarding the patient and aircraft scheduling portion of the AE system. It presents a discussion of literature

illustrating aeromedical evacuation, a brief description and comparison of recent research efforts, and an overview of mixed integer linear programming (MILP) considerations.

AE System Background

A report submitted by Battelle Columbus Division entitled, *Concept Development for an Aeromedical Data Acquisition and Communication System (AMDACS)*, provides an outstanding overview of the AE process (7). It gives insight into medical regulation procedures, patient categories, and transportation requirements. It also gives a detailed definition of the four-echelon medical care system, the National Disaster Medical System, and various agencies' responsibilities in those systems (7:10-40). Most of the data concerning the medical component of the AE system used in this research was obtained from this report.

Information regarding specific AMC actions was obtained from, *Annex Q to AMC Omnibus OPLAN*. The purpose of this annex is:

> to provide the concept of operations, assign tasks, and furnish generic guidance for the development of the medical care system in support of the operations envisaged in regional OPLANs,... no-plan contingency operations, and [also] military operations short of war (5:Q-2).

This regulation details definitions and duties of AMC agencies, including the Aeromedical Evacuation Control Center (AECC), which is now part of the GPMRC. The GPMRC is responsible for scheduling all AE aircraft and monitoring patient movement (5:Q-19). With this basic information, current research efforts concerning the AE system could be explored.

Recent Research Efforts

Many recent studies in the area of aeromedical evacuation concentrate on optimizing the aircraft fleet designated to fly this mission by minimizing the total number of aircraft required or minimizing patient wait time using existing fleets. Prior to 1986, the Military Airlift Command (MAC), now AMC, AE plan was based on the use of C-141 and C-9 aircraft for the bulk of the AE mission during wartime (8:7). Because of concern about the C-141 fleet's ability to accomplish this task along with its other wartime requirements, MAC sponsored a study by a team at the Air Force Institute of Technology (AFIT). In 1985, Captain Joseph P. Alfano and Captain John C. O'Neill published a thesis, *Wartime CONUS Casualty Distribution System Using Dedicated CRAF Aircraft*, which focused on evaluating the capability of Civil Reserve Air Fleet (CRAF) aircraft to fulfill aeromedical evacuation requirements during a large-scale contingency (1). The team built a computer simulation which modeled casualty transportation requirements. A few key aspects in their model development were:

- The number of casualties to be evacuated, which was supplied by MAC, represented unclassified casualty estimates for the first 60 days of an intense European conventional war (1:21).

- Because no CRAF aircraft were designated prior to the study, the capacities of the aircraft modeled had to be estimated (1:22).

- The assumed patient distribution system was a hub design. Patients would initially be flown from an airport in the communications zone (COMMZ) of the combat theater, to a hub airport in the CONUS. If beds were available in a nearby hospital which was capable of treating the casualties, they would be ground transported to that hospital. If no beds were available, the patients would be air evacuated to a spoke airport where hospitals near that spoke airport could handle the casualty. In the research, only one hub airport, Dover AFB, DE, and its spoke system were modeled (1:31).

- Patient-to-bed assignments were not made until the casualty arrived at Dover AFB. At that time, the simulation algorithm would look for an open hospital bed in the Dover AFB area. If no beds were available, the algorithm would then look for a bed at the spoke airports' hospitals (1:33).

Using the patients' average time in system, Alfano and O'Neill concluded a fleet of four CRAF aircraft, with a capacity of 175 litter patients each, could feasibly accomplish the aeromedical evacuation requirements (1:76).

As a result of studies like Alfano and O'Neill's, in 1986 the Secretary of the Air Force authorized the use of a dedicated aeromedical segment of the CRAF (8:7). In 1993, following development and testing of new equipment, 14 aircraft had been committed to this mission with an expected addition of 30 more. These aircraft would be available if activated by the Secretary of Defense (17:16). Because of this added capability, AMC requested studies to evaluate the optimal size of the CRAF fleet required to carry out the AE mission. One of these studies was a thesis conducted by Major Charles R. Wolfe, entitled *The Use of Simulation to Evaluate Strategic Aeromedical Evacuation Policy and Planning* (18). Wolfe used simulation and modeling techniques similar to those of Alfano and O'Neill. The key difference between Wolfe's study and the one conducted by Alfano and O'Neill concerns the measure of effectiveness. Along with mean time in system, Wolfe used a variety of values, including aircraft utilization rates and percentage of missions delayed due to lack of available aircraft. His stated goal in doing this was "[if] given a representative scenario, to take a macro approach to determine the major drivers affecting strategic AE and the fundamental relationship between these factors" (18:3-2).

Other differences between Alfano and O'Neill's study and Wolfe's include:

- Wolfe's study spanned 180 days and used a two theater scenario, assuming simultaneous conflicts in Southwest Asia and the Far East. This changed not only the number of casualties but also the route structure of the returning aircraft (18:3-3).

- Using updated information regarding the CRAF fleet, Wolfe's simulation was modeled with 45 Boeing 767 series 200 aircraft with a capacity of 102 patients each (18:3-3).

- Although the hub design was still assumed, six hub airfields were modeled (18:3-6).

- Patient-to-bed assignments were assumed to be made while the patient was in the combat theater. No scheduling criteria for patients was given (18:3-16).

- Wolfe did not consider the in-CONUS distribution of patients, but he did track the bed availability for individual hospitals (18:3-8).

As stated above, the purpose of Wolfe's study was not, starting with given parameters, to find an optimal number of beds, aircraft or airports. The study focused rather on identifying the most influential factors effecting the policy decisions regarding the AE process. The concluding remarks were a discussion of the model itself and how it could be used to benefit an agency responsible for the patient and aircraft scheduling of the wartime AE system. The study contained a recommendation to model the entire scheduling process within a single computer package to allow for real-time scheduling during a contingency (10:64-68).

During the period of Wolfe's analysis, Major Mike Loftus studied the AE system using an entirely different approach (10). In his thesis entitled, *Patient Scheduling and Aircraft Routing for Strategic Aeromedical Evacuation*, Loftus used mathematical programming techniques in developing an approach to first assign patients to AE aircraft and then to assign these aircraft to CONUS destinations (10:vii). Loftus was not primarily interested in looking at how many aircraft were needed to support the AE mission.

Instead, the study was designed to take the resources available, and through scheduling, minimize the patients' wait in the system before being evacuated to the CONUS (10:7).

Important conditions built into the model by Loftus were:

- The use of a worst-case scenario of 2000 patients per day (10:34).

- One COMMZ airport and 13 hub airports were modeled (10:7,82).

- The CRAF fleet would be made up of 44 Boeing 767s with capacities of 100 litter patients each (10:8). On any single day, ten of these aircraft would arrive at the COMMZ airport (10:34).

- Like Wolfe, Loftus assumed patient-to-bed assignments were made in-theater and in-CONUS patient distribution was disregarded (10:7-8).

The significant differences between Loftus' study and the two previously discussed studies are the methods of mathematical formulation and the planning horizon used. As noted in the discussions of the work performed by Alfano and O'Neill, and Wolfe, they employed simulation techniques which modeled the AE system over long time spans of 60 and 180 days, respectively. The outputs of these simulations were then analyzed using statistical analysis software packages to determine the values of, and relationships between, the factors that would determine required aircraft fleet size. These studies were designed to evaluate information for managing the macro-level of the AE system, specifically aircraft procurement. Therefore, simulation methods were ideal because of their ability to describe overall system dynamics (1),(18).

In Loftus' study, mathematical programming techniques were used. These methods included mixed integer programming, network flow programming, scheduling, and the use of heuristics (10:vii). This allowed the construction of good, feasible patient and aircraft schedules for a given level of input. As a result of the computer time required

to solve his model, one of Loftus' concluding recommendations was to further develop the model concentrating on methods to accelerate solution time. The following section discusses some important aspects and definitions involved in possible mathematical programming approaches to the AE system.

MILP Solution Considerations

MILPs formulated like the problem above are not *difficult* problems to solve. However, finding an optimal solution *quickly* to every instance of the problem may take a great deal of time. This section discusses some general considerations in reducing the solution time of this research's problem, including an overview of complexity theory and heuristics as they apply to this research.

A scientific discipline called complexity theory studies the tractability of problems and corresponding solution algorithms. Parker and Rardin state the goals of complexity theory are to "broadly classify problems and algorithms according to their convenience for solution by digital computers" (14:3). Problems are categorized in terms of mathematical functions expressing computation *time* as a function of problem size. For example, polynomial time means that as the number of variables in the problem increases, the number of steps required to solve the problem increases polynomially as a function of problem size. Although complexity theory applies to decision problems with yes or no answers, the MILP can be restated as a decision problem (14:8). Nemhauser and Wolsey describe a few of the problem classes as (13:119-137):

- *P* - problems that can be solved by *polynomial* time algorithms

- *NP* (Nondeterministically Polynomial) - a class of problems for which an algorithm exists that will verify the feasibility of a feasible instance in polynomial time.

- *NP*-Hard - the set of problems to which all *NP* problems polynomially reduce

- *NP*-Complete - problems which are contained in both *NP* and *NP*-Hard and are considered the most difficult to solve.

MILPs similar to the one in this research are of the class *NP*-Complete (15:83-84). Of course, these problems can be solved by total enumeration of all feasible points, but for this research's problem, total enumeration would be extremely expensive computationally. Some solution approaches to NP-Complete problems search for good, near-optimal, rather than optimal solutions. These approaches are called heuristics, and although they may even find the optimal solution, they cannot *prove* that a solution is optimal in polynomial time. Loftus' research described one such heuristic approach to the aeromedical evacuation problem (10).

Like many *NP*-Complete problems, many instances of the problem in this research can be solved quickly using a branch and bound algorithm. In this problem, an instance is characterized by the number of variables, their coefficients, and the right-hand-sides of the constraints. If one of these changes even by a small amount, the previously solvable instance may become much more difficult to solve. Because of this, the heuristic framework used in this research is to employ a branch and bound method, using CPLEX, to quickly find a *good, feasible* solution. If time permits, the CPLEX solver can be used to search for the *optimal* solution. A more detailed description of the solution process is given in Chapter 4 of this research. In Chapter 3, the foundation for this process is presented.

III. Methodology

General

The primary purpose of this research is to develop a means for GPMRC personnel to forecast and plan the efficient use of aeromedical airlift resources during a contingency. This chapter discusses the methodology used in the solution process of this research. The initial section contains a general overview of problem information and motivation for the mathematical formulation of the problem. The next sections cover the user-defined parameters, present the mixed integer linear program formulation, and give an overview of the solution process.

To solve this problem, the AE system is modeled as a mixed integer linear program. Even though input information is projected, and therefore uncertain, the short time horizon (2-5 days) generally leads to fairly accurate projections. Also, the product of this research is a *forecast* of aircraft requirements for these projected inputs. Because of these conditions, much of the stochastic nature of the problem is reduced, so a deterministic approach seems appropriate.

The goal of this deterministic approach, represented as the objective function in a mixed integer linear program (MILP), is to minimize the number of dedicated C-141 aircraft required to evacuate the patients during the projected time span. While some of the individual aircraft may not be completely filled by patients, it is nevertheless necessary to model the aircraft as integers in the formulation. This stipulation states that a single aircraft may only fly one route, so it may only pick up patients at one airport in the AOR

and drop off patients at a single destination hub airport in the CONUS. Using this assumption, the problem must be modeled with integer variables representing aircraft.

Another goal is to evacuate all patients. Because of this, the objective function includes penalties for not evacuating patients. These penalties force the model to evacuate as many patients as possible. Since reducing the number of non-preferred hubs is simply a secondary goal, an order of magnitude difference between the penalties assigned to the aircraft and the non-preferred hubs was used. GPMRC has asserted that this relationship is reasonable (11).

Although a patient would not fly on more than one aircraft, the integer requirement for the variables representing patient-to-aircraft assignments has been relaxed. After restricting the aircraft to integer values and then choosing a feasible set of those variables, the problem then appears to take the form of a network flow problem which resulted in integer value solutions for the patient variables in all test cases. The relaxation of the integer requirements of patients greatly reduces the number of integer variables in the formulation, and thus reduces the time required to solve the problem. The use of continuous variables in this fashion leads to an MILP.

The objective function also uses penalties to discourage the use of destination airports not preferred by the user. Preferred airports can be those closest to the AOR or airports with the most available beds or airports preferred for other reasons.

As with any real system, a variety of limitations are placed upon the AE system. In this research's MILP formulation, the constraints are used to model these restrictions as well as to impose requirements on the solution. Basically the constraints force all patients

to be flown or placed in non-evacuated status, and bound the number of available aircraft and beds, the patients flown to the hub airports, and patients flown on an aircraft.

User-Defined Values

Clearly the number of patients requiring transport, the number of available beds, and the number and type of aircraft available would change regularly throughout a contingency's duration. This section describes what values are input by the user, and how those values are used to formulate the objective function and constraints described above.

The values input by the user can be separated into three classes: those used to assist input and output analysis, those used to define parameters, and those that affect the number of variables and constraints in the problem. The first class of inputs simply assign names to the AOR airports, hub airports, and the patient classes.

The second class of inputs affect the coefficients of variables in the constraints, the coefficients of variables in the objective function, or the value of the right-hand-sides of the constraints. These inputs do not change the number of variables in the model. These inputs are:

- hub preferences,

- number of patients per patient class at each AOR airport,

- number of beds per patient class at each hub airport,

- number and capacity of aircraft for each type,

- maximum number of patients a hub airport can handle daily.

The final class of user inputs directly determine the number of variables and are the most critical when considering the size of the problem and the time required to solve.

These inputs are the number of hub and AOR airports and the number of patient classes. The number of aircraft types also affects the number of variables; however, in this research, only three aircraft types are considered: CRAF aircraft, dedicated C-141s, and retrograde C-141s. There is an aircraft variable for every AOR-Aircraft Type-Hub combination, and since only these variables are restricted to be integer valued in this research, the number of aircraft variables is the most important factor in determining the complexity of the problem.

Mixed Integer Linear Program Formulation

As noted by Nemhauser and Wolsey, "In integer programming, formulating a 'good' model is of crucial importance to solving the model" (13:14). Many of the variable definitions are given by the problem itself, but the selection of the variable restrictions and the form of the constraints can vary between modelers. In the following formulation, three groups of variables are defined. After defining the variables, the mathematical formulation of the model is presented.

Variable Definition. Each of the variable groups representing aircraft, patients evacuated, and patients in a non-evacuated status has associated indices to completely described the variables in that group. Table 3.1 lists the indices used in this formulation. HUB is used in the remainder of the thesis to refer to the CONUS airports.

Table 3.1 Variable Indices

Index	Description
i	Patient injury class identifier
j	Individual aircraft type identifier
k	AOR (departure) airport identifier
l	CONUS (HUB) airport identifier

The number of patient injury classes, HUB airports and AOR airports are defined by the user. Because of variable length restrictions, the indices can range from one to 99 for both injury classes and HUB airports, and from one to nine for AOR airports. For aircraft types, the index j always ranges from one to three for the above mentioned aircraft types and has values as shown in Table 3.2.

Table 3.2 Aircraft Indices

j	Aircraft Type
1	CRAF Boeing 767s
2	Retrograde C-141s
3	Dedicated C-141s

The aircraft variable, AC_{jkl}, is equal to the number of aircraft type j flying the route from AOR airport k to HUB airport l. There are j times k times l aircraft variables. A patient variable, P_{ijkl}, equals the number of patients at AOR airport k with injury class i that will be evacuated on an aircraft of type j to HUB airport l. The final group of variables, $NOEV_{ik}$, represents the number of patients at AOR airport k with injury class i that are not evacuated.

Several parameters are also used in the formulation. They are:

$PTot_{ik}$ = Number of patients of injury class i awaiting transportation from AOR airport k.

$ACTot_j$ = Number of aircraft type j available for use in the time-frame of the model.

$ACCap_j$ = Patient capacity of type j aircraft.

$BedTot_{il}$ = Number of injury class i designated beds available at HUB airport l.

$Flow_l$ = Flow-through capacity, maximum number of patients HUB airport l can handle daily.

Model Formulation.
Define the following sets:

$\quad$ I $\quad = \{i \mid 1 \le i \le$ user-input number of patient classes, $i \in Z^+\}$

$\quad$ J $\quad = \{j \mid 1 \le j \le 3, j \in Z^+\}$

$\quad$ K $\quad = \{k \mid 1 \le k \le$ user-input number of AOR airports, $k \in Z^+\}$

$\quad$ L $\quad = \{l \mid 1 \le l \le$ user-input number of Hub airports, $l \in Z^+\}$

$\quad$ N $\quad = \{l \mid l$ is not a preferred Hub airport, $l \in Z^+\}$

where Z^+ represents the positive integers.

Objective Function (when dedicated C-141s are available):

$$\text{Min} \quad \sum_{k \in K} \sum_{l \in N} (1.1) AC_{3kl} + \sum_{k \in K} \sum_{l \notin N} AC_{3kl} + \sum_{j=1,2} \sum_{k \in K} \sum_{l \in N} (0.1) AC_{jkl} + \sum_{k \in K} \sum_{i \in I} 100(NOEV_{ik}) \qquad (3\text{-}1)$$

Subject To:

$$\sum_{j \in J} \sum_{l \in L} P_{ijkl} + NOEV_{ik} = PTot_{ik} \qquad \forall \ i \in I, \ k \in K \qquad (3\text{-}2)$$

$$\sum_{k \in K} \sum_{l \in L} AC_{jkl} \le ACTot_j \qquad \forall \ j \in J \qquad (3\text{-}3)$$

$$\sum_{i \in I} P_{ijkl} - (ACCap_j) * AC_{jkl} \le 0 \qquad \forall \ j \in J, \ k \in K, \ l \in L \qquad (3\text{-}4)$$

$$\sum_{k \in K} \sum_{j \in J} P_{ijkl} \le BedTot_{il} \qquad \forall \ i \in I, \ l \in L \qquad (3\text{-}5)$$

$$\sum_{k \in K} \sum_{i \in I} \sum_{l \in L} P_{ijkl} \le Flow_l \qquad \forall \ l \in L \qquad (3\text{-}6)$$

$$AC_{jkl} \{Z^+\}, \ P_{ijkl}, NOEV_{ik} \ge 0 \quad \forall \ i \in I, j \in J, k \in K, l \in L \qquad (3\text{-}7)$$

The objective function, Equation (3-1), minimizes the number of dedicated C-141s flown, the number of patients not evacuated, and the number of aircraft which fly to non-preferred HUB airports. This minimization is accomplished by use of four sets of terms. The first term is the number of dedicated C-141s that are flown to non-preferred HUB airports. A penalty of 0.1 is assessed for each aircraft using a non-preferred HUB. The second term is the number of dedicated C-141s required in the solution which are not flown to a non-preferred HUB airport. The third term is a penalty for CRAF aircraft and retrograde C-141s using non-preferred HUB airports. If hub preferences are <u>not</u> used, all hubs are considered preferred, the set N is empty, and the coefficients of the first and third terms are zero. The last term represents the penalties for not evacuating patients from their respective AOR airports.

The constraints enforce the requirements on the problem. The constraints in Equation (3-2) force the model to assign every patient either to an aircraft for evacuation or to non-evacuated status. The constraints in Equation (3-3) are the upper bounds on the number of each type of aircraft. The constraints represented by Equation (3-4) state that the total number of patients flying on an aircraft type-route combination must be less than the capacity of that aircraft. Equation (3-5) ensures the number of patients in an injury class evacuated to the individual HUB airports is less than the number of available beds designated for that injury class at that HUB airport. The last set of constraints, Equation (3-6), restrict the total number of patients sent to a HUB airport to be less than that airport's flow-though capacity.

Equation (3-7) states that the value of each aircraft type-route combination in the

solution must be a non-negative integer and that the number of patients assigned to the

aircraft type-route combinations or not evacuated must be greater than or equal to zero.

No Available Dedicated C-141s. If no dedicated C-141s are available, an

alternate objective function is constructed. Equation (3-8) shows this alternate objective

function. Because AE CRAF aircraft are equipped specifically to handle patient

evacuation, and to minimize the strain on the overall military airlift system, the alternate

objective function is formulated to reduce the number of retrograde C-141s required. The

four terms in Equation (3-8) are basically the same as those in Equation (3-1). The first

two terms represent the number of retrograde C-141s using preferred and non-preferred

hub airports, respectively. The third term is the number of CRAF aircraft flying to non-

preferred hubs. The last term of Equation (3-8) is identical to the last term of Equation

(3-1).

$$\text{Min} \quad \sum_{k \in K} \sum_{l \in N} (1.1) AC_{2kl} + \sum_{k \in K} \sum_{l \notin N} AC_{2kl} + \sum_{k \in K} \sum_{l \in N} (0.1) AC_{1kl} + \sum_{k \in K} \sum_{i \in I} 100(NOEV_{ik}) \qquad (3\text{-}8)$$

The constraints used when no dedicated C-141s are available are identical to those

described in the previous section. The only change is that the set J now only contains two

types of aircraft, CRAF and retrograde C-141s, so the index j ranges from 1 to 2.

Solution Process

The solution approach of this research is a two-stage process. A flow chart of the process is shown in Figure 3.1 on the next page. The initial step is the input, by the user, of data defining the problem. The user defined input values are described in Chapter 3. The result of this input step is the construction of a mathematical programming system (MPS) file to be read by the solution program. This file is basically a representation of the mathematical model. The FORTRAN code for the MPS file building program (mpsbldr.f) and the solution program (solver.f) are located in the Appendices A and B, respectively. User instructions for both executable programs are located in Appendix C.

After the MPS file has been built, the second program reads this file and begins the optimization. The user then inputs whether or not hub preferences were used in building the data set, and decides how many nodes will initially be used as the stopping criteria. If an optimal solution is found, or the problem is found to be infeasible, the program stops and outputs the solution. If this first set of nodes is exhausted and no optimal solution is found, the user may either accept the given *best* solution or continue the process by increasing the number of nodes to be searched. The best solution, if integer, will also be output.

Because the objective of the research is to quickly produce a *forecast* of the number of aircraft required, and the *projected* routes and patient loads of those aircraft, no search is made for alternate optimal solutions. When an integer solution is found, all nodes with equal or greater possible solutions are fathomed. As soon as a *proven* optimal

solution is discovered, the solution process is ended. Therefore, alternate optimal

solutions may exist but will not be displayed.

A number of test cases were run using the programs and solution process

described in this chapter. In Chapter 4, results of the process as applied to these test cases

is presented.

Figure 3.1 Solution Process Flow Chart

```
                    ╭──────────╮
                    │ Data     │
                    │ Input    │
                    ╰──────────╯
                         │
                         ▼
                   ┌──────────┐
                   │ Construct│
                   │ MILP     │
                   └──────────┘
                         │
                         ▼
                   ┌──────────┐
                   │ Read     │
                   │ MILP     │
                   └──────────┘
                         │
                         ▼
                   ┌──────────┐
                   │ Solve Using│
         ┌────────▶│ Node Limit │
         │         └──────────┘
         │              │
         │              ▼
         │         ┌──────────┐
         │         │ Analyze  │
         │         │ Solution │
         │         └──────────┘
         │              │
         │              ▼
   ┌──────────┐       ◇ Does
   │ Update   │◀─Yes──◇ Node Limit
   │ Node Limit│      ◇ Need
   └──────────┘       ◇ Updating
                         │
                         No
                         ▼
                   ┌──────────┐
                   │ Stop With│
                   │ "Best" Found│
                   │ Solution │
                   └──────────┘
```
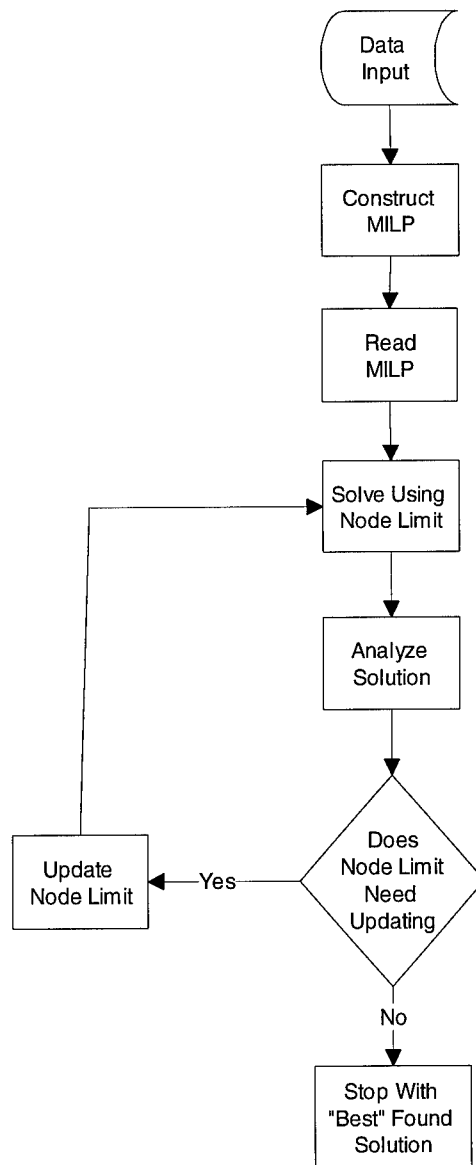
IV. Model and Result Analysis

The purpose of this chapter is to describe the analysis of test results for the solution methodology developed in Chapter 3. First, a description of the test data used for the analysis is presented. This is followed by a discussion of the solution method's efficiency. The last section of the chapter is a review of the solution results for various test cases.

Test Cases

The test data used for this analysis is contained in Appendix D. The total number of patients and available beds was held constant in all data sets. In the test data only six out of the 13 patient classes listed in Chapter 1 are used. Increasing the number of patient classes will increase the number of patient variables, but since these variables are not restricted to integer values, the increased number of variables should not greatly affect the overall solution time. Four AOR airports are generally used, but when less than four are used, the number of patients from the deleted AOR airports are evenly distributed to the remaining AOR airports. Test case 1 is used as the base case, and generally only one area is different between the base case and each of the other test cases. For example, preferences for hubs one through five were added to test case 2, while the values for all other variables and parameters are identical to those of the base case. The test cases, their differences from the base case, and solution results are presented in Table 4.1.

Table 4.1 Test Data Sets

Test Case	Differences From Base Case	Node Limit	Time	Obj. Function Value *=proven optimum	Node
1	None	10000	4 sec	*5	17
2	hubs 1-5 preferred	10000	5 sec	506.2	32
			7 sec	6.2	70
			22.5 min		node limit
		30000	37 min	5.4	18344
			42.5 min	5.2	21461
			61 min		node limit
3	hubs 5-9 preferred	10000	5 sec	6.2	35
			18.6 min	5.3	3981
			24.3 min	5.2	7341
			26.8 min		node limit
4	hubs 2,4,6,8 preferred	10000	5 sec	5.4	17
			27 min		node limit
		30000	74 min		node limit
5	A/C totals changed	10000	6 sec	2903	34
			7 sec	503	46
			11 sec	3	82
			8.7 min		node limit
		100000	84 min		node limit
6	A/C totals changed 0 Dedicated C-141s used	10000	3 sec	16	36
			5 sec	15	104
		100000	88 min		node limit
7	3 AORs used	10000	4 sec	6	43
			12	*5	194
8	2 AORs used	10000	2 sec	*5	20
9	1 AOR used	10000	2 sec	*5	22
10	Flow-through of 300	10000	4 sec	*5	32
11	Flow-through of 400	10000	3 sec	1606	49
			5 sec	6	82
			10 sec	*5	174
12	Flow-through of 500	10000	2 sec	1606	44
			3 sec	6	47
			3 min	*5	5585
13	Flow-through of 1000	10000	3 sec	6	20
			7 sec	*5	130
14	Alternate flow-throughs of 250 & 500 used	10000	3 sec	6	24
			7 sec	*5	130
15	Alternate flow-throughs of 250 & 500 used and alt. Hub preferences	10000	4 sec	6.2	18
			2 min	5.2	2345
			8.5 min		node limit

Table 4.1 Test Data Sets (continued)

Test Case	Differences From Base Case	Node Limit	Time	Obj. Function Value *=proven opt.	Node
16	Flow-throughs of 500 used and hubs 1-5 preferred	10000	5 sec	6.0	50
			15 sec	5.1	184
			7.5 min	5.0	9372

Solution Method Analysis

As discussed in Chapter 2, different instances of the same basic MILP model can have vastly different solution times. Table 4.1 presents an excellent example of this phenomenon, as shown in the wide range of solution times for the test cases run on a Sun SPARC station 2. While the optimal solution was found quickly for some test cases, in many test cases no proven optimal solution was found. Test case 5 shows that even if a one hundred thousand node limit is used, an optimal solution is not assured. An optimal solution is found whenever the number of aircraft equals the minimum number of aircraft required by the relaxed LP solution. The minimum number of aircraft is the smallest integer value greater than or equal to the number of aircraft in the relaxed LP solution. In most of the test cases, a solution with one dedicated C-141 more than the minimum established by the relaxed LP solution was found in less than 10000 nodes. This is shown in Table 4.2. This means for the majority of the test cases, the primary goal of this research has been met: take user input values and quickly produce a feasible schedule of aircraft to routes and patients to aircraft that minimizes the number of dedicated C-141s used.

Table 4.2 Relaxed LP and "Best" IP Objective Function Values

Test Case	Relaxed LP Solution	Best IP Solution in 10000 nodes
1	4.1	5
2	4.219	6.2
3	4.219	5.2
4	4.469	5.4
5	1.433	3
6	12.8167	15
7	4.0222	5
8	4.1	5
9	4.1	5
10	4.1	5
11	4.1	5
12	4.1	5
13	4.1	5
14	4.1	5
15	4.219	5.2
16	4.1	5.0

One area which appears to have a large effect on the solution time is the use of penalties to reduce the number of non-preferred hub airports. When hub preferences are used (test cases 2, 3, 4, 15 and 16) there is an additional 0.1 penalty assessed to aircraft route combinations that use a non-preferred hub airport as its destination. This is shown in Equation (3-1), which is repeated below as Equation (4-1). Given this penalty, any *better* integer solution for the remaining nodes must have an objective function value at least 0.1 less than the current best integer solution. Because CPLEX does not inherently *know* this, an objective function value difference of 0.099 is set in the solution program. This difference forces the fathoming of nodes with objective function values greater than the best integer solution's objective function value minus 0.099. While this action reduces the number of remaining nodes, in this research's test cases with hub preferences, the solution times were still relatively high compared to cases without hub preferences.

$$\text{Min } \sum_{k\in K}\sum_{l\in N}(1.1)AC_{3kl} + \sum_{k\in K}\sum_{l\notin N}AC_{3kl} + \sum_{j=1,2}\sum_{k\in K}\sum_{l\in N}(.1)AC_{jkl} + \sum_{k\in K}\sum_{i\in I}100(NOEV_{ik}) \qquad (4\text{-}1)$$

For test cases 1 and 5-14, where hub preferences are not used, the first and third terms have zero coefficients since the set N is empty; thus, the 0.1 penalty is not seen in the objective function. Therefore, any *better* integer solution for the remaining nodes of these instances of the problem must have an objective function value at least 1.0 less than the current best integer solution, and an objective function value difference of 0.99 is used by the solution program. As shown in Table 4.1, the solution times were significantly lower for the test cases where hub preferences were not used, and in only two cases, test case 5 and test case 6, was an optimal solution not found. In each of the test cases for which an optimal solution was found, the solution of the relaxed LP was greater than the objective function value of the optimal integer solution minus 0.99. This relationship was the principal reason for the short solution time for these cases.

The notable exceptions in the reduced solution times for cases where no hub preferences were used are test cases 5 and 6. In all other test cases where hub preferences were used, the difference between the objective function values of the relaxed LP solution and the best IP solution was less than 0.99. When the 0.99 objective value difference was subtracted, all remaining nodes were then fathomed. In test case 5; however, the objective function value of the relaxed LP was 1.4333 while the objective function value of the best integer solution found was 3.0. Therefore, the difference between them, 1.5777, was greater than the set 0.99 objective value difference. Because of this, the solver could not immediately fathom all remaining nodes and prove that 3.0 was the optimal solution. The objective function value difference eliminated the remaining nodes with subproblem

objective function values greater than 2.001. However, in this test case there were still many nodes left with objective function values between 1.4333 and 2.001. This caused test case 5's solution time to be much greater than the other cases not using hub preferences. In fact, the optimal solution was not found, or proved optimal, in one hundred thousand nodes. The corresponding analysis of test case 6 reveals the same problem, with a relaxed LP objective function value of 12.8167, and a best integer objective function value of 15.

Test Results Analysis

The "Objective Function Value" column of Table 4.1 lists objective function values found for the test cases at the respective nodes. The final entry for each test case is the *best* solution found using the specified node limit. Examples of the complete solution files for test cases 1, 2, and 9, including values for the non-zero variables, are contained in Appendix E. This section reviews the objective function values for the various test cases and analyzes important factors that appear to contribute to the solution differences or similarities between cases.

The base case, test case 1, has an optimal solution with the objective function value of 5.0, which means that a minimum of five dedicated C-141s are required to fulfill the AE requirements for the given input. Since no hub preferences were entered, all hubs are considered preferred, and no penalties are assessed for any routes.

Hub Preferences and Flow-Through Capacities. Hub preferences are used in test cases 2-4 and 15-16. The solution process for these cases took longer to find good, and then better, feasible values than test case 1. But, solutions using only five dedicated

C-141s for each of these test cases were found in the initial 10000 node limit with the exception of test case 2 which took over 18000 nodes. These objective function values are an expected result since the primary goal is to minimize these aircraft and the secondary goal is to minimize the number of non-preferred hubs used.

Flow-through capacities are the maximum number of patients that can be handled on a daily basis at a hub airport. The base case value of 250 for all hubs was changed in test cases 10-16. Test cases 10-13 use increasing capacities of 300, 400, 500, and 1000. In each case an optimal solution of five dedicated C-141s was found. This shows that given the test data used, the number of aircraft has little apparent dependence on the number of patients a hub airport can handle.

Increasing flow-through capacities with hub preferences can change the objective function values, as noted in test cases 15 and 16. Test case 15 differs from test case 2 only with respect to the increased flow-through capacity of 500 patients at the non-preferred hub airports, 6 through 9. The objective function value of the best integer solution for test cases 2 and 15 was 5.2. An analysis of the results shows only two aircraft were sent to non-preferred hubs. But, when all flow-through capacities were raised to 500 patients, as in test case 16, no aircraft were sent to the non-preferred airports.

Number of AORs. Test cases 1, 7, 8, and 9 used decreasing numbers of AOR airports. Initially, it was assumed that streamlining the patient flow by reducing the number of AORs would decrease the number of aircraft required by causing fewer aircraft to fly at less than full capacity. As seen in Table 4.1, each of these cases required five dedicated C-141s. However, the solution shown in Table 4.3 for test case 9, where only

one AOR was used, reveals that only nine out of the available 10 retrograde C-141s were used. As a product of the model formulation, all of these aircraft should be used prior to using the dedicated C-141s. Examining the solution for test case 9 (see Appendix E) reveals that the use of the additional 60 patient capacity of a retrograde C-141 would still have left more than four dedicated C-141 loads of patients to be evacuated, and thus five dedicated C-141s are required. The number of patients to be transported from the chosen AOR airports obviously will influence the total aircraft used. As shown in test case 9, the types of aircraft required can also be affected.

Table 4.3 Aircraft Fleets

Test Case	CRAF Aircraft Available/Used	Retrograde C-141s Available/Used	Dedicated C-141s Available/Used	Total Available/Used
1-4,7-8,10-16	4/4	10/10	6/5	20/19
5	4/4	14/14	3/3	21/21
6	6/6	18/15	0	24/21
9	4/4	10/9	6/5	20/18

Aircraft Fleet Make-Up. Table 4.3 shows that in test cases 5 and 6, the number of each type of aircraft were altered from the original fleet in test case 1. Because of the varying capacities between aircraft types, the fleet make-up is an important factor in the total number of aircraft used. As described in the previous section, the number of AORs and the total number of patients can also influence the total number of aircraft used, and to a smaller degree the aircraft types used.

Chapter 4 has reviewed the results for 16 test cases. Chapter 5 presents the conclusions from this research and recommendations for future studies.

V. Conclusions and Recommendations

The primary objective of this research was to develop a tool for USTRANSCOM GPMRC to use in forecasting and scheduling aircraft and patients effectively for the strategic aeromedical evacuation *lift-bed* process. The programs and solution process produced during this research effort provide an effective means to accomplish these tasks. The purpose of this chapter is to review major conclusions and to provide recommendations for further research.

Conclusions

This research effort investigated the strategic aeromedical evacuation *lift-bed* process and provides a method for forecasting required airlift and scheduling aircraft and patients to be evacuated. Even though the AE process is only a single subset of the entire combat effort, it is an extremely important one. However, it is desired to make the AE system have the smallest impact on the other portions of the overall effort. To reduce the impact, specifically on the strategic airlift assets, it is necessary to minimize the resources taken from the airlift system. An effective means of reducing the number of aircraft used by the AE system is to optimally allocate the aircraft in the AE system, while efficiently transporting the injured combatants back to the United States' medical facilities.

The solution process and associated computer programs developed in this research provide timely, suitable forecasts of the needed airlift resources and the aircraft-to-airport and patients-to-aircraft assignments. These forecasts minimize the important military

resources required, specifically the number of C-141s that must be dedicated to the AE mission. The process provided good, and in most cases optimal, solutions for the number of dedicated C-141s necessary to complete the evacuation of the patients. The time required to obtain these solutions vary between each problem instance but are well within the time frame specified by the users. When the process has been terminated by the user, the output from the process is easily recovered in a solution file that shows the best forecast found and specifically lists the number of dedicated C-141s required and their routes.

The process developed is user-friendly and allows an interactive exchange of input values. Questions to the user require either yes-or-no responses, or an input value such as the number of available retrograde C-141s. Additionally, the programs developed can be run on equipment that GPMRC currently has access to.

Recommendations for Further Research

The solution process developed in this research effort uses CPLEX to solve the mixed integer linear program that is constructed. To solve MILPs, CPLEX employs a branch-and-bound method. There is currently work being done to refine a solution process which uses a hybrid of branch-and-bound and cutting plane methods to solve MILPs more quickly. This research's solution process may be improved if this new solution approach is applied to it.

The solution process will find a best or optimal solution and then remove from consideration all other solutions with the same objective function value. Therefore, the user is not able to search for alternate optimal solutions. Means to search for alternate

optimal solutions in MILPs have been developed, and could be appended to this solution

process. This would permit the exploration of other optimal schedules.

While the process was designed to find quick, optimal solutions for the forecasting

and scheduling functions of GPMRC, it is not limited to these uses. This solution process

can be integrated with simulated patient movement requirements and recovery rates for an

extended contingency. Impacts of a variety of planned resource set-ups can then be

investigated, like overall aircraft fleet requirements, and the number of AOR and hub

airports.

APPENDIX A: FORTRAN Code for mpsbldr.f

Presented in this Appendix is the FORTRAN Code used to build the executable file. The program, after being properly compiled, will prompt the user for the required input data. The output from this program will be a data file, containing a collection of the input data, and an MPS file to be read by solver.f. The data and MPS filenames will be the filename entered as input, followed by ".dat", and ".mps", respectively.

```
*      This program is designed to retrieve input values from the screen,
*      then have the user make any necessary changes.
*      It will then build the .MPS file to be used by the solver.f.
*****************************************************************************
         INTEGER       I,J,K,L,PCLTOT(8,13),CLASS,PATCOUNT(4),BEDS(10,6)
         INTEGER       NUMHUB,NUMAOR,NUMPAT,NUMPT(8),ACCAP(4)
         INTEGER       TOTAC,M,F,Q,EQCONST,LTCONST,NUMPCL,PCL,CONSTNUM
         INTEGER       NUMAC(3),ACTYPES, A, B, FLOWCAP(10), HUBPREF(10)
         CHARACTER*15  CLSNAME(13), AORNAME(10), HUBNAME(10)
         CHARACTER*3   YESORNO, PREFYORN,HUBSYN
         CHARACTER*8   INFILE, FILNAM
         CHARACTER*12  MPSFIL12, DATFIL12
         CHARACTER*11  MPSFIL11, DATFIL11
         CHARACTER*10  MPSFIL10, DATFIL10
         CHARACTER*9   MPSFIL9,  DATFIL9
         CHARACTER*8   MPSFIL8,  DATFIL8
         CHARACTER*7   MPSFIL7,  DATFIL7
         CHARACTER*6   MPSFIL6,  DATFIL6
         CHARACTER*5   MPSFIL5,  DATFIL5
         REAL PENALTY
*****************************************************************************
*                     OPENING THE MPS FILE                       *
         FILNAM = '        '
         WRITE (*,'(//A)') 'Enter the Filename for this data set.'
         WRITE (*,'(A)') '!!Note: Filename must be 8 letters or less!!'
           READ (*,'(A)')   INFILE
*
         IF (INFILE(2:8) .EQ. ' ') THEN
           MPSFIL5 = INFILE(1:1)//'.mps'
             OPEN(UNIT=10, FILE=MPSFIL5)
             WRITE(10, '(2A)') 'NAME     ',MPSFIL5
           DATFIL5 = INFILE(1:1)//'.dat'
             OPEN(UNIT=20, FILE=DATFIL5)
           FILNAM(8:8) = INFILE(1:1)
         ELSEIF (INFILE(3:8) .EQ. ' ') THEN
           MPSFIL6 = INFILE(1:2)//'.mps'
             OPEN(UNIT=10, FILE=MPSFIL6)
             WRITE(10, '(2A)') 'NAME     ',MPSFIL6
           DATFIL6 = INFILE(1:2)//'.dat'
             OPEN(UNIT=20, FILE=DATFIL6)
           FILNAM(7:8) = INFILE(1:2)
         ELSEIF (INFILE(4:8) .EQ. ' ') THEN
           MPSFIL7 = INFILE(1:3)//'.mps'
             OPEN(UNIT=10, FILE=MPSFIL7)
```

```
          WRITE(10, '(2A)') 'NAME    ',MPSFIL7
        DATFIL7 = INFILE(1:3)//'.dat'
         OPEN(UNIT=20, FILE=DATFIL7)
        FILNAM(6:8) = INFILE(1:3)
      ELSEIF (INFILE(5:8) .EQ. ' ') THEN
        MPSFIL8 = INFILE(1:4)//'.mps'
         OPEN(UNIT=10, FILE=MPSFIL8)
          WRITE(10, '(2A)') 'NAME    ',MPSFIL8
        DATFIL8 = INFILE(1:4)//'.dat'
         OPEN(UNIT=20, FILE=DATFIL8)
        FILNAM(5:8) = INFILE(1:4)
      ELSEIF (INFILE(6:8) .EQ. ' ') THEN
        MPSFIL9 = INFILE(1:5)//'.mps'
         OPEN(UNIT=10, FILE=MPSFIL9)
          WRITE(10, '(2A)') 'NAME    ',MPSFIL9
        DATFIL9 = INFILE(1:5)//'.dat'
         OPEN(UNIT=20, FILE=DATFIL9)
        FILNAM(4:8) = INFILE(1:5)
      ELSEIF (INFILE(7:8) .EQ. ' ') THEN
        MPSFIL10 = INFILE(1:6)//'.mps'
         OPEN(UNIT=10, FILE=MPSFIL10)
          WRITE(10, '(2A)') 'NAME    ',MPSFIL10
        DATFIL10 = INFILE(1:6)//'.dat'
         OPEN(UNIT=20, FILE=DATFIL10)
        FILNAM(3:8) = INFILE(1:6)
      ELSEIF (INFILE(8:8) .EQ. ' ') THEN
        MPSFIL11 = INFILE(1:7)//'.mps'
         OPEN(UNIT=10, FILE=MPSFIL11)
          WRITE(10, '(2A)') 'NAME    ',MPSFIL11
        DATFIL11 = INFILE(1:7)//'.dat'
         OPEN(UNIT=20, FILE=DATFIL11)
        FILNAM(2:8) = INFILE(1:7)
      ELSE
        MPSFIL12 = INFILE//'.mps'
         OPEN(UNIT=10, FILE=MPSFIL12)
          WRITE(10, '(2A)') 'NAME    ',MPSFIL12
        DATFIL12 = INFILE//'.dat'
         OPEN(UNIT=20, FILE=DATFIL12)
        FILNAM = INFILE
      ENDIF
************************************************
      J=0
      K=0
      L=0
      NUMHUB=9
      NUMAOR=4
      NUMPCL=6
      WRITE (20,'(A)') '**********************************************'
      WRITE (20,'(2A)') 'THIS IS THE DATA USED TO CREATE ',
C          FILNAM//'.mps'
      WRITE (20,'(A/)') '**********************************************'
      WRITE (*,'(//A,I2)') 'The Number of AOR Airports is currently = ',
C                                               NUMAOR
      WRITE (*,'(A,I2)')'The Number of HUB Airports is currently = ',
C                                               NUMHUB
      WRITE (*,'(2A,I2/)')'The Number of Patient Classes is currently',
C                                  ' = ',    NUMPCL
      WRITE (*,'(A)')   'Would you like to change any of these values?'
```

```
         WRITE (*,'(A)')   'Y or N, then RETURN'
         READ (*,'(A)')    YESORNO
         IF (YESORNO .EQ. 'Y' .OR. YESORNO .EQ. 'YES' .OR. YESORNO .EQ.
      c    'y' .OR. YESORNO .EQ. 'yes' .OR. YESORNO .EQ. 'Yes') THEN
           WRITE (*,'(/A)')   'Please enter the number of AOR Airports'
             READ (*,'(I2)')   NUMAOR
           WRITE (*,'(/A)')   'Please enter the number of HUB Airports'
             READ (*,'(I2)')   NUMHUB
           WRITE (*,'(/A)')   'Please enter the number of Patient Classes'
             READ (*,'(I2)')   NUMPCL
         ENDIF
*
         WRITE (20,'(A,I2)') 'Number of AOR Airports used =     ', NUMAOR
         WRITE (20,'(A,I2)') 'Number of HUB Airports used =     ', NUMHUB
         WRITE (20,'(A,I2/)')'Number of Patient Classes used = ', NUMPCL
         WRITE (*,'(A)')   'Would you like to enter AOR names?'
         WRITE (*,'(A)')   'Y or N, then RETURN'
         READ (*,'(A)')    YESORNO
*
         IF (YESORNO .EQ. 'Y' .OR. YESORNO .EQ. 'YES' .OR. YESORNO .EQ.
      c    'y' .OR. YESORNO .EQ. 'yes' .OR. YESORNO .EQ. 'Yes') THEN
         WRITE (20,'(A)')   'The AOR names are:'
           DO 6 K=1,NUMAOR,1
             WRITE (*,'(/A,I2,A)')   'AOR # ',K,'s name is:'
               READ (*,'(A)') AORNAME(K)
         WRITE (20,'(A,I2,2A)')   'AOR # ',K,' is ',AORNAME(K)
6          CONTINUE
         ENDIF
*
         WRITE (*,'(A)')   'Would you like to enter HUB names?'
         WRITE (*,'(A)')   'Y or N, then RETURN'
         READ (*,'(A)')    YESORNO
*
         IF (YESORNO .EQ. 'Y' .OR. YESORNO .EQ. 'YES' .OR. YESORNO .EQ.
      c    'y' .OR. YESORNO .EQ. 'yes' .OR. YESORNO .EQ. 'Yes') THEN
         WRITE (20,'(A)')   'The HUB names are:'
           DO 7 L=1,NUMHUB,1
             WRITE (*,'(/A,I2,A)')   'HUB # ',L,'s name is:'
               READ (*,'(A)') HUBNAME(L)
         WRITE (20,'(A,I2,2A)')   'HUB # ',L,' is ',HUBNAME(L)
7          CONTINUE
         ENDIF
*
         DO 8 L=1,NUMHUB,1
           HUBPREF(L) = 0
8        CONTINUE
*
         WRITE (*,'(A)')   'Would you like to enter HUB preferences?'
         WRITE (*,'(A)')   'Y or N, then RETURN'
         READ (*,'(A)')    HUBSYN
*
         IF (HUBSYN .EQ. 'Y' .OR. HUBSYN .EQ. 'YES' .OR. HUBSYN .EQ.
      c    'y' .OR. HUBSYN .EQ. 'yes' .OR. HUBSYN .EQ. 'Yes') THEN
             WRITE (20,'(A)') ' '
           DO 9 L=1,NUMHUB,1
             WRITE (*,'(/A,I2,A)')   'Do you PREFER to use HUB #',
      c              L,'?    Y or N'
             READ (*,'(A)') PREFYORN
```

A-3

```
          IF (PREFYORN.EQ.'Y' .OR. PREFYORN.EQ.'YES' .OR. PREFYORN
   c      .EQ.'y' .OR. PREFYORN.EQ.'yes' .OR. PREFYORN.EQ.'Yes') THEN
              HUBPREF(L)=1
              WRITE (20,'(A,I2,A)') 'HUB # ',L,' was preferred.'
          ENDIF
   9      CONTINUE
   *
       ELSE
         WRITE (20,'(A)')  'HUB preferences were not used!'
         WRITE (20,'(A)') 'All HUBs are considered preferred.'
        DO 11 L=1,NUMHUB,1
            HUBPREF(L)=1
   11     CONTINUE
       ENDIF
   *
       DO 10 K=1,NUMAOR,1
          DO 15 PCL=1,NUMPCL,1
            PCLTOT(K,PCL)=0
   15       CONTINUE
          PATCOUNT(K)=0
          NUMPT(K)=0
   10     CONTINUE
   *
       CLSNAME(1)  = 'GEN. MEDICAL'
       CLSNAME(2)  = 'PSYCHIATRIC'
       CLSNAME(3)  = 'SURGICAL'
       CLSNAME(4)  = 'ORTHOPEDIC'
       CLSNAME(5)  = 'SPINAL'
       CLSNAME(6)  = 'BURN'
       CLSNAME(7)  = 'NEUROSURGICAL'
   *
       IF (NUMPCL .GT. 7) THEN
          DO 25 I=8,NUMPCL,1
            WRITE (*,'(//A)')'PLEASE INPUT NEXT INJURY CLASS NAME.'
            READ (*,'(A)') CLSNAME(I)
   25     CONTINUE
       ENDIF
   *
       WRITE (*,'(//A)')'THESE ARE THE CURRENT INJURY CLASSES:'
       WRITE (20,'(/A)')'THESE ARE THE INJURY CLASSES:'
       DO 20 I=1,NUMPCL,1
       WRITE (*,'(A)') CLSNAME(I)
       WRITE (20,'(A)') CLSNAME(I)
   20     CONTINUE
   *
       WRITE (*,'(//A)')'Do you want to use test PATIENT values?'
       READ (*,'(A)')    YESORNO
       IF (YESORNO .EQ. 'N' .OR. YESORNO .EQ. 'NO' .OR. YESORNO .EQ.
   c      'n' .OR. YESORNO .EQ. 'no' .OR. YESORNO .EQ. 'No') THEN
          WRITE (*,'(//A)')   '****************************************'
          WRITE (*,'(/2A)')   'Please enter the number of Patients ',
   c                  'at each AOR Airport, per patient class.'
          WRITE (*,'(/A)')    '****************************************'
   *
       DO 30 K=1,NUMAOR,1
          WRITE (*,'(//A)')   '****************************************'
             WRITE (*,'(/A,I2)')   'AOR AIRPORT NUMBER ',K
          DO 35 PCL=1,NUMPCL,1
```

```
              WRITE (*,'(/3A,I2,A)')  'The Number of ', CLSNAME(PCL),
     c          'Patients at AOR #',K,' ='
              READ (*,'(I3)')  PCLTOT(K,PCL)
35        CONTINUE
              WRITE (*,'(/A)')  '***************************************'
30      CONTINUE
      ELSE
      WRITE (*,'(/A)')  'Using Test Patient Values!!'
*********************************************************************
*              TEST NUMBERS OF PATIENTS AND BEDS              *
      PCLTOT(1,1)=43
      PCLTOT(1,2)=10
      PCLTOT(1,3)=143
      PCLTOT(1,4)=120
      PCLTOT(1,5)=10
      PCLTOT(1,6)=3
      PCLTOT(1,7)=0
      PCLTOT(2,1)=43
      PCLTOT(2,2)=10
      PCLTOT(2,3)=143
      PCLTOT(2,4)=120
      PCLTOT(2,5)=5
      PCLTOT(2,6)=5
      PCLTOT(2,7)=0
      PCLTOT(3,1)=46
      PCLTOT(3,2)=11
      PCLTOT(3,3)=161
      PCLTOT(3,4)=132
      PCLTOT(3,5)=11
      PCLTOT(3,6)=4
      PCLTOT(3,7)=0
      PCLTOT(4,1)=37
      PCLTOT(4,2)=5
      PCLTOT(4,3)=167
      PCLTOT(4,4)=120
      PCLTOT(4,5)=13
      PCLTOT(4,6)=7
      PCLTOT(4,7)=0
      ENDIF
*********************************************************************
*              COMBINING PATIENT IF < 4 AORS USED              *
      IF (NUMAOR .EQ. 1) THEN
        DO 40 B=1,NUMPCL,1
          DO 45 A=2,4,1
            PCLTOT(1,B)=PCLTOT(1,B)+PCLTOT(A,B)
45        CONTINUE
40      CONTINUE
      ELSEIF (NUMAOR .EQ. 2) THEN
        DO 50 B=1,NUMPCL,1
          PCLTOT(1,B) = PCLTOT(1,B) + PCLTOT(3,B)
        . PCLTOT(2,B) = PCLTOT(2,B) + PCLTOT(4,B)
50      CONTINUE
      ELSEIF (NUMAOR .EQ. 3) THEN
        DO 60 B=1,NUMPCL,1
          DO 65 A=1,3,1
            PCLTOT(A,B) = PCLTOT(A,B) + ABS(PCLTOT(4,B)/3)
65        CONTINUE
60      CONTINUE
```

```fortran
      ENDIF
*
*     Writing bed info to .dat file
      WRITE (20,'(/A)') 'Patient totals for (AOR#,PatClass#) are:'
      DO 70 A=1,NUMAOR,1
        DO 75 B=1,NUMPCL,1
      WRITE(20,'(A,I1,A,I2,A,I4)')'PCLTOT(',A,',',B,')= ', PCLTOT(A,B)
75      CONTINUE
70      CONTINUE
***************************************************************
*                       BEDS/HUBS                          *
      WRITE (*,'(//A)')'Do you want to use test BED values?'
      READ (*,'(A)')    YESORNO
      IF (YESORNO .EQ. 'N' .OR. YESORNO .EQ. 'NO' .OR. YESORNO .EQ.
     c    'n' .OR. YESORNO .EQ. 'no' .OR. YESORNO .EQ. 'No') THEN
        WRITE (*,'(//2A)')   '***************************************',
        WRITE (*,'(/2A)')'Please enter the number of Beds Available',
     c                   ' at each HUB Airport, per patient class.'
        WRITE (*,'(/2A)')   '***************************************',
*
        DO 80 L=1,NUMHUB,1
          WRITE (*,'(//A)')   '***************************************'
            WRITE (*,'(/A,I2)')  'HUB AIRPORT NUMBER ',L
          DO 85 PCL=1,NUMPCL,1
            WRITE (*,'(/3A,I2,A)')  'The Number of ', CLSNAME(PCL),
     c         'Beds Available at HUB #',L,' ='
              READ (*,'(I3)')  BEDS(L,PCL)
85        CONTINUE
          WRITE (*,'(/A)')   '***************************************'
80      CONTINUE
      ELSE
      WRITE (*,'(/A)')   'Using Test Bed Values!!'
*     LA/TUCSON/LUKE
      BEDS(1,1)=4704+169+371
      BEDS(1,2)=1343+66+110
      BEDS(1,3)=3327+82+472
      BEDS(1,4)=1102+35+47
      BEDS(1,5)=567+9+21
      BEDS(1,6)=473+8+5
*     SF/FTLEWIS/PORTLAND
      BEDS(2,1)=897+473+213
      BEDS(2,2)=421+180+64
      BEDS(2,3)=1481+450+327
      BEDS(2,4)=358+264+81
      BEDS(2,5)=117+81+15
      BEDS(2,6)=56+77+12
*     BOSTON/NHAMPTON/ALBANY
      BEDS(3,1)=3593+233+180
      BEDS(3,2)=303+6+104
      BEDS(3,3)=2612+132+181
      BEDS(3,4)=368+66+10
      BEDS(3,5)=97+4+1
      BEDS(3,6)=55+5+1
*     CHARLOTTE/FTJACKSON/FTGORDON/FTBRAGG/CHARLESTON
      BEDS(4,1)=527+420+274+1621+212
      BEDS(4,2)=344+110+212+533+78
      BEDS(4,3)=370+242+169+1632+277
      BEDS(4,4)=102+149+150+714+62
```

```
          BEDS(4,5)=41+24+29+142+3
          BEDS(4,6)=14+27+14+135+12
*       DENVER/HILL/WICHITA/ALBQ/FTBLISS
          BEDS(5,1)=603+171+52+118+603
          BEDS(5,2)=190+72+30+27+115
          BEDS(5,3)=732+186+46+123+539
          BEDS(5,4)=437+32+42+36+389
          BEDS(5,5)=26+20+0+0+27
          BEDS(5,6)=3+17+1+5+3
*       PHIL/SYRACUSE/BUFFALO/PITTSBURGH/NORFOLK/WASHDC
          BEDS(6,1)=4209+219+813+1366+1201+1604
          BEDS(6,2)=1286+31+128+110+516+357
          BEDS(6,3)=3367+47+671+946+1061+1403
          BEDS(6,4)=784+16+202+134+434+711
          BEDS(6,5)=228+4+25+30+22+170
          BEDS(6,6)=102+38+17+10+46+23
*       HOUSTON/NEWORLEANS/L.ROCK/SHREVEPORT/OKCITY/CARSWELL/SANANTONIO
          BEDS(7,1)=1211+2323+80+229+227+683+1150
          BEDS(7,2)=246+635+10+101+52+147+291
          BEDS(7,3)=1205+1271+83+106+422+502+428
          BEDS(7,4)=402+325+86+32+120+174+138
          BEDS(7,5)=5+79+0+0+1+52+9
          BEDS(7,6)=26+23+0+4+0+0+35
*       ATLANTA/B'HAM/ORLANDO/J'VILLE/JACKSON/MILLINGTON/K'VILLE/N'VILLE
          BEDS(8,1)=706+514+875+456+336+427+206+468
          BEDS(8,2)=84+71+154+193+166+160+57+199
          BEDS(8,3)=467+487+702+393+315+248+209+490
          BEDS(8,4)=309+56+201+355+23+59+64+161
          BEDS(8,5)=3+12+57+54+16+25+9+13
          BEDS(8,6)=35+9+50+57+8+4+1+5
*       CHICAGO/CLEVELAND/MINNEAPOLIS/DESMOINES/INDIANAPOLIS
*       SCOTT/LEAVONWORTH/LEXINGTON/ALLENPARK/OFFUT/WRIGHT-PATT
          BEDS(9,1)=2204+344+88+65+250+665+222+261+592+583+537
          BEDS(9,2)=568+197+149+27+77+130+76+92+154+104+155
          BEDS(9,3)=2651+287+338+84+84+1216+382+395+514+126+880
          BEDS(9,4)=547+103+73+12+33+276+84+192+130+178+318
          BEDS(9,5)=88+41+27+4+8+0+24+5+29+14+120
          BEDS(9,6)=95+33+11+3+3+0+6+1+26+16+96
          ENDIF
*
*       Writing bed info to .dat file
        WRITE (20,'(/A)') 'BED totals for (HUB#,PatClass#) are:'
        DO 87 A=1,NUMHUB,1
           DO 88 B=1,NUMPCL,1
           WRITE (20,'(A,I2,A,I2,A,I4)') 'BEDS(',A,',',B,') = ',
     C             BEDS(A,B)
88      CONTINUE
87      CONTINUE
*******************************************************************
        DO 89 L=1,NUMHUB,1
           FLOWCAP(L) = 250
89      CONTINUE
*
        WRITE (*,'(//2A/A)')'The MAX Number of Patients for Each HUB',
     c  ' is 250 Patients.', 'Would You Like to Change This Value?'
        READ (*,'(A)')    YESORNO
        IF (YESORNO .EQ. 'Y' .OR. YESORNO .EQ. 'YES' .OR. YESORNO .EQ.
     c     'y' .OR. YESORNO .EQ. 'yes' .OR. YESORNO .EQ. 'Yes') THEN
```

```fortran
            WRITE (*,'(/2A)') 'Please enter the Max number of patients',
     C                      ' allowed to transit each HUB Airport.'
*

            DO 90 L=1,NUMHUB,1
              WRITE (*,'(/2A,I2,A)')  'The MAX Number of Patients',
     C           ' Allowed to Transit HUB # ',L,' ='
                READ (*,'(I4)')  FLOWCAP(L)
              WRITE (20,'(2A,I2,A,I4)')  'The MAX Number of Patients',
     C           ' Allowed to Transit HUB # ',L,' = ',FLOWCAP(L)
90      CONTINUE
*

        ELSE
          WRITE (*,'(/2A)')  'Using Flow-through capacity of 250 ',
     C             'patients at each HUB!!'
          WRITE (20,'(/2A)')  'Using Flow-through capacity of 250 ',
     C             'patients at each HUB!!'
        ENDIF
*********************************************************************
        DO 100 K=1,NUMAOR,1
        DO 105 CLASS=1,NUMPCL,1
          NUMPT(K)=NUMPT(K)+PCLTOT(K,CLASS)
105     CONTINUE
        NUMPAT=NUMPAT+NUMPT(K)
100     CONTINUE
*

        ACCAP(1)=100
        ACCAP(2)=60
        ACCAP(3)=90
        NUMAC(1)=4
        NUMAC(2)=10
        NUMAC(3)=6
***********************************************
        WRITE (*,'(//A,I2,A)') 'There are currently ',NUMAC(1),
     C                     '  CRAF Aircraft,'
        WRITE (*,'(A,I2,A)') '                        ',NUMAC(2),
     C                     '  Retrograde C-141s,'
        WRITE (*,'(A,I2,A)') '                        ',NUMAC(3),
     C                     '  Dedicated C-141s.'
        WRITE(*,'(/2A)')'Would you like to change the number',
     C           ' of aircraft?'
        WRITE (*,'(A)')   'Y or N, then RETURN'
        READ (*,'(A)')    YESORNO
        IF (YESORNO .EQ. 'Y' .OR. YESORNO .EQ. 'YES' .OR. YESORNO .EQ.
     C    'y' .OR. YESORNO .EQ. 'yes' .OR. YESORNO .EQ. 'Yes') THEN
        WRITE(*,'(//2A)')'Please enter the Number of',' CRAF Aircraft.'
          READ (*,'(I2)')   NUMAC(1)
         WRITE (*,'(/2A)')  'Please enter the Number of',
     C                      ' Retrograde C-141s.'
          READ (*,'(I2)')   NUMAC(2)
         WRITE (*,'(/2A)')  'Please enter the Number of',
     C                      ' Dedicated C-141s.'
          READ (*,'(I2)')   NUMAC(3)
        ENDIF
*

        WRITE (20,'(/A,I2,A)') 'There are ',NUMAC(1),'  CRAF Aircraft,'
        WRITE (20,'(A,I2,A)')  '              ',NUMAC(2),
     C                     '  Retrograde C-141s,'
        WRITE (20,'(A,I2,A)')  '              ',NUMAC(3),
```

```
      C                            '  Dedicated C-141s.'
**************************************************
      WRITE (*,'(//2A,I3)') 'The Capacity of CRAF Aircraft ',
      C                               'is currently = ', ACCAP(1)
       WRITE (*,'(2A,I3)') 'The Capacity of Retrograde C-141s ',
      C                               'is currently = ', ACCAP(2)
       WRITE (*,'(2A,I3)')'The Capacity of Dedicated C-141s ',
      C                               'is currently = ', ACCAP(3)
*
       WRITE (*,'(/A)')   'Would you like change to these values?'
       WRITE (*,'(A)')   'Y or N, then RETURN'
       READ (*,'(A)')    YESORNO
       IF (YESORNO .EQ. 'Y' .OR. YESORNO .EQ. 'YES' .OR. YESORNO .EQ.
      C    'y' .OR. YESORNO .EQ. 'yes' .OR. YESORNO .EQ. 'Yes') THEN
        WRITE (*,'(//A)')   'Please enter the CRAF Aircraft Capacity'
          READ (*,'(I2)')    ACCAP(1)
         WRITE (*,'(/A)')   'Please enter the Retrograde C-141 Capacity'
          READ (*,'(I2)')    ACCAP(2)
         WRITE (*,'(/A)')   'Please enter the Dedicated C-141 Capacity'
          READ (*,'(I2)')    ACCAP(3)
       ENDIF
*
       WRITE (20,'(/2A,I3)') 'The Capacity of CRAF Aircraft ',
      C                               'is      = ', ACCAP(1)
       WRITE (20,'(2A,I3)') 'The Capacity of Retrograde C-141s ',
      C                               'is = ', ACCAP(2)
       WRITE (20,'(2A,I3)')'The Capacity of Dedicated C-141s ',
      C                               'is   = ', ACCAP(3)
*************************************************
       ACTYPES=3
       IF (NUMAC(3) .EQ. 0) ACTYPES=2
       TOTAC=0
       DO 110 Q=1,ACTYPES,1
         TOTAC=TOTAC+NUMAC(Q)
110    CONTINUE
       M=0
       F=0
*****************************************************************
*                 WRITING THE MPS FILE                         *
*****************************************************************
*                 NAMING SENSE OF CONSTRAINTS                  *
*****************************************************************
       WRITE(10, '(A)') 'ROWS'
       WRITE(10, '(A)') ' N  obj'
*
       EQCONST=NUMAOR*NUMPCL
       DO 135 I=1,EQCONST,1
         WRITE(10, '(A,I4.4)') ' E    c',I
135      CONTINUE
*
       LTCONST=ACTYPES+(NUMAOR*ACTYPES*NUMHUB)+(NUMHUB*NUMPCL)+NUMHUB
       DO 140 Q=1,LTCONST,1
         CONSTNUM=EQCONST+Q
         WRITE(10, '(A,I4.4)') ' L    c',CONSTNUM
140      CONTINUE
*
       WRITE(10, '(A)') 'COLUMNS'
       WRITE(10, '(A)') '    MARK0000  ''MARKER''           ''INTORG'''
```

```
*****************************************************************
*                    AIRCRAFT VARIABLES                        *
*****************************************************************
*           FOR VARIABLE AC (J,K,L)
      DO 200 K=1,NUMAOR,1
       DO 210 J=1,ACTYPES,1
      DO 220 L=1,NUMHUB,1
********************************************************
*      Intergerize Coefficients if no Hubs are Preferred
      IF (HUBSYN.NE.'Y'.AND. HUBSYN.NE.'YES'.AND. HUBSYN
   c   .NE.'y'.AND. HUBSYN.NE.'yes'.AND. HUBSYN.NE.'Yes') THEN
        IF (ACTYPES .EQ. 3) THEN
          IF (J.EQ.3) THEN
            WRITE(10, '(A,3I2.2,A,I7,A,I4.4,I7)')  ' AC',J,K,L,
   C        '        obj ',1,'      c',NUMAOR*NUMPCL+J,1
            WRITE(10, '(A,3I2.2,A,I4.4,I7)')   ' AC',J,K,L,
   C        '        c',ACTYPES+NUMAOR*NUMPCL+
   C      NUMHUB*(J-1)+ACTYPES*NUMHUB*(K-1)+L,(-1*ACCAP(J))
*
          ELSE
            WRITE(10, '(A,3I2.2,A,I4.4,I7,A,I4.4,I7)') ' AC',
   C          J,K,L, '          c',NUMAOR*NUMPCL+J,1,'     c',
   C          ACTYPES+NUMAOR*NUMPCL+NUMHUB*(J-1)
   C          +ACTYPES*NUMHUB*(K-1)+L,(-1*ACCAP(J))
          ENDIF
*
        ELSE IF (ACTYPES .EQ. 2) THEN
          IF (J.EQ.2) THEN
            WRITE(10, '(A,3I2.2,A,I7,A,I4.4,I7)')   ' AC',J,K,L,
   C        '        obj ',1,'    c',NUMAOR*NUMPCL+J,1
            WRITE(10, '(A,3I2.2,A,I4.4,I7)')   ' AC',J,K,L,
   C        '        c',ACTYPES+NUMAOR*NUMPCL+
   C      NUMHUB*(J-1)+ACTYPES*NUMHUB*(K-1)+L,(-1*ACCAP(J))
*
          ELSE
            WRITE(10, '(A,3I2.2,A,I4.4,I7,A,I4.4,I7)') ' AC',
   C          J,K,L, '          c',NUMAOR*NUMPCL+J,1,'     c',
   C          ACTYPES+NUMAOR*NUMPCL+NUMHUB*(J-1)
   C          +ACTYPES*NUMHUB*(K-1)+L,(-1*ACCAP(J))
          ENDIF
        ENDIF
*
      ELSE
        IF (ACTYPES .EQ. 3) THEN
          PENALTY = .1
          IF (J.EQ.3) PENALTY=1
          IF (J.EQ.3 .AND. HUBPREF(L).EQ.0) PENALTY=1.1
*
          IF (J.EQ.3 .OR. HUBPREF(L).EQ.0) THEN
            WRITE(10, '(A,3I2.2,A,F7.2,A,I4.4,I7)')   ' AC',J,K,L,
   C        '        obj ',PENALTY,'     c',NUMAOR*NUMPCL+J,1
*
            WRITE(10, '(A,3I2.2,A,I4.4,I7)')   ' AC',J,K,L,
   C        '         c',ACTYPES+NUMAOR*NUMPCL+
   C        NUMHUB*(J-1)+ACTYPES*NUMHUB*(K-1)+L,(-1*ACCAP(J))
*
          ELSE
          WRITE(10, '(A,3I2.2,A,I4.4,I7,A,I4.4,I7)') ' AC',
```

```
      C          J,K,L, '              c',NUMAOR*NUMPCL+J,1,'      c',
      C              ACTYPES+NUMAOR*NUMPCL+NUMHUB*(J-1)
      C              +ACTYPES*NUMHUB*(K-1)+L,(-1*ACCAP(J))
            ENDIF
*
        ELSE IF (ACTYPES .EQ. 2) THEN
           IF (J.EQ.2) PENALTY=1
           IF (J.EQ.2 .AND. HUBPREF(L).EQ.0) PENALTY=1.1
*
           IF (J.EQ.2 .OR. HUBPREF(L).EQ.0) THEN
             WRITE(10, '(A,3I2.2,A,F7.2,A,I4.4,I7)')    ' AC',J,K,L,
      C        '        obj ',PENALTY,'       c',NUMAOR*NUMPCL+J,1
             WRITE(10, '(A,3I2.2,A,I4.4,I7)')    ' AC',J,K,L,
      C        '        c',ACTYPES+NUMAOR*NUMPCL+
      C      NUMHUB*(J-1)+ACTYPES*NUMHUB*(K-1)+L,(-1*ACCAP(J))
*
           ELSE
             WRITE(10, '(A,3I2.2,A,I4.4,I7,A,I4.4,I7)')  ' AC',
      C          J,K,L, '         c',NUMAOR*NUMPCL+J,1,'      c',
      C             ACTYPES+NUMAOR*NUMPCL+NUMHUB*(J-1)
      C             +ACTYPES*NUMHUB*(K-1)+L,(-1*ACCAP(J))
           ENDIF
         ENDIF
        ENDIF
220   CONTINUE
210   CONTINUE
200   CONTINUE
      WRITE(10, '(A)') '    MARK0001  ''MARKER''          ''INTEND'''
***********************************************************************
*                  PATIENT VARIABLES                          *
***********************************************************************
*          FOR VARIABLE (K,I,J,L)
      DO 400 K=1,NUMAOR,1
       DO 410 I=1,NUMPCL,1
*
        WRITE(10, '(A,2I2.2,A,I7,A,I4.4,I7)')
      C  '  NOEV',K,I,'          obj',100,'         c',(K-1)*NUMPCL+I,1
*  MUST BE ASSIGNED ONLY 1 ROUTE OR NONEVAC
        DO 420 J=1,ACTYPES,1
         DO 430 L=1,NUMHUB,1
          WRITE(10,'(A,I1.1,3I2.2,A,I4.4,I7,A,I4.4,I7)')
      C         ' P',K,I,J,L,'         c',(K-1)*NUMPCL+I,1,
*  NUMBER SENT TO A HUB MUST BE LESS THEN HUB'S
*             FLOW-THROUGH CAPACITY
      C  '             c',NUMAOR*NUMPCL+ACTYPES+NUMAOR*ACTYPES*NUMHUB+
      C                   NUMPCL+(NUMHUB-1)*NUMPCL+L,1
*
*  NUMBER ON PLANE MUST BE < AIRPLANE CAPACITY
          WRITE(10, '(A,I1.1,3I2.2,A,I4.4,I7,A,I4.4,I7)')
      C       ' P',K,I,J,L,'         c',
      C     NUMAOR*NUMPCL+ACTYPES+(K-1)*NUMHUB*ACTYPES+
      C       NUMHUB*(J-1)+L,1,
*  NUMBER TO HUB MUST BE < AVAILABLE BEDS FOR EACH PAT. INJURY CLASS
      C  '             c',NUMAOR*NUMPCL+ACTYPES+NUMAOR*ACTYPES*NUMHUB+
      C                   I+(L-1)*NUMPCL,1
430     CONTINUE
420     CONTINUE
410    CONTINUE
```

A-11

```
400    CONTINUE
*
*       WRITE(10, '(A)')'    MARK0001  ''MARKER''       ''INTEND'''
****************************************************************
*                              RHS                            *
****************************************************************
       WRITE(10, '(A)') 'RHS'
       DO 500  K=1,NUMAOR,1
        DO 510 I=1,NUMPCL,1
        WRITE(10, '(A,I4.4,I7)') '    rhs    c',(K-1)*NUMPCL+I,
     C                    PCLTOT(K,I)
510    CONTINUE
500    CONTINUE
*
       DO 520   Q=1,ACTYPES,1
       WRITE(10, '(A,I4.4,I7)') '    rhs    c',NUMAOR*NUMPCL+Q,
     C      NUMAC(Q)
520    CONTINUE
*
       DO 550   L=1,NUMHUB,1
        DO 570   I=1,NUMPCL,1
         WRITE(10, '(A,I4.4,I7)') '    rhs    c',NUMAOR*NUMPCL+ACTYPES+
     C        NUMAOR*ACTYPES*NUMHUB+(L-1)*NUMPCL+I,BEDS(L,I)
570    CONTINUE
*
*    FLOW-THROUGH RESTRICTIONs
         WRITE(10, '(A,I4.4,I7)') '    rhs    c',NUMAOR*NUMPCL+ACTYPES+
     C        NUMAOR*ACTYPES*NUMHUB+NUMPCL+(NUMHUB-1)*NUMPCL+L,FLOWCAP(L)
550    CONTINUE
****************************************************************
*                             BOUNDS                          *
****************************************************************
       WRITE(10, '(A)') 'BOUNDS'
*       FOR VARIABLE AC (J,K,L)
       DO 600 J=1,ACTYPES,1
         DO 610 K=1,NUMAOR,1
          DO 620 L=1,NUMHUB,1
          WRITE(10, '(A,3I2.2,I7)')  ' UP bnd      AC',
     C                                J,K,L,NUMAC(J)
620       CONTINUE
610       CONTINUE
600    CONTINUE
*
*        FOR VARIABLE PT (K,I,J,L)
       DO 700 K=1,NUMAOR,1
        DO 710 I=1,NUMPCL,1
          WRITE(10, '(A,2I2.2,I7)')  ' UP bnd      NOEV',
     C      K,I,PCLTOT(K,I)
         DO 720 J=1,ACTYPES,1
          DO 730 L=1,NUMHUB,1
          WRITE(10, '(A,I1.1,3I2.2,I7)')  ' UP bnd      P',
     C      K,I,J,L,PCLTOT(K,I)
730       CONTINUE
720       CONTINUE
710     CONTINUE
700    CONTINUE
*
       WRITE(10, '(A)') 'ENDATA'
```

```
         CLOSE(UNIT=10, STATUS='KEEP')
         CLOSE(UNIT=20, STATUS='KEEP')
*
         END
```

APPENDIX B: FORTRAN Code for solver.f

        This appendix contains the FORTRAN code for the solution program. After being properly compiled, it will prompt the user for the input MPS filename, whether or not hub preferences were used, and the initial and continued node limits. The output will be a solution file "<filename>.sol", where the values of the non-zero variables are stored, and a log file which provides solution process information. In addition, this information will be output to the screen during the running of the program.

```
*    Program to solve MPS files by calling CPLEX library routines
*    using Sun Fortran to C interface capability. The Fortran routines
*    call intermediate C routines that call the CPLEX library routines.
*    This approach does require using the CPLEX supplied intermediate
*    C routines (INTERMED.C).
*    All of the intermediate routines are functions, and all arguments
*    are passed by reference.
*    Interface definitions: The following routine 'impsmrd' will call
*    two CPLEX library routines, 'mpsmread' and 'loadmprob'.  This
*    avoids the difficulty of passing pointers in Fortran.
*
        external sscrin   !$pragma C (sscrin)
        external slogfo   !$pragma C (slogfo)
        external smint    !$pragma C (smint)
        external impsmrd  !$pragma C (impsmrd)
        external imipopt  !$pragma C (imipopt)
        external gmar     !$pragma C (gmar)
        external gmac     !$pragma C (gmac)
        external gitcm    !$pragma C (gitcm)
        external gstat    !$pragma C (gstat)
        external gmx      !$pragma C (gmx)
        external gmobj    !$pragma C (gmobj)
        external gndc     !$pragma C (gndc)
        external ifreems  !$pragma C (ifreems)
        external ifreep   !$pragma C (ifreep)
        external icptyp   !$pragma C (icptyp)
        external isolut   !$pragma C (isolut)
        external iopt     !$pragma C (iopt)
        external sndlim   !$pragma C (sndlim)
        external gbobj    !$pragma C (gbobj)
        external scutup   !$pragma C (scutup)
        external impswr   !$pragma C (impswr)
        external gcname   !$pragma C (gcname)
        external sadvin   !$pragma C (sadvin)
        external gcutup   !$pragma C (gcutup)
        external sobjdf   !$pragma C (sobjdf)
        external svarsel  !$pragma C (svarsel)
*  Constants
        integer          macsz
        parameter        (macsz = 4000)
*  Declarations
        double precision    obj
```

```
          double precision     mipobj
*   Solution value arrays must be at least as large as problem
*   row and column counts
          double precision     mipx(0:macsz), pobjdf
          character*8          store(0:macsz)
*   Local variables
          integer       status,numvar, j, gstatus
          integer       nodelo,nodehi,nodelim,newnode
          integer       cstorsz, surplus, begin, end
          integer       namelen,advance, diflo, difhi
          integer       printint, pintlo, pinthi
          integer       vselect, vsello, vselhi
          CHARACTER*8   INFILE,CONTINUE, HUBPREF
          CHARACTER*12  MPSFIL12,SOLFILE12
          CHARACTER*11  MPSFIL11,SOLFILE11
          CHARACTER*10  MPSFIL10,SOLFILE10
          CHARACTER*9   MPSFIL9,SOLFILE9
          CHARACTER*8   MPSFIL8,SOLFILE8
          CHARACTER*7   MPSFIL7,SOLFILE7
          CHARACTER*6   MPSFIL6,SOLFILE6
          CHARACTER*5   MPSFIL5,SOLFILE5
*   Functions
          integer       sscrin, slogfo, smint, impsmrd, imipopt
          integer       gmar, gmac, gitcm, gstat gcname, gcutup
          integer       gmx, gmobj, gndc, ifreems, icptyp, sadvin
          integer       isolut, iopt, sndlim, gbobj, scutup,
          integer       impswr, pmac, sobjdf, svarsel, gcname
********************************
*   Send log output to stdout
          status = sscrin (1)
          status = slogfo ('airevac.log')
          printint = 1000
          pintlo = 100
          pinthi = 5000
*
          WRITE (*,'(//A)') 'Enter the Filename to be solved'
          WRITE (*,'(A)') '!!Note: Filename must be 8 letters or less!!'
            READ (*,'(A)')  INFILE
          IF (INFILE(2:8) .EQ. ' ') THEN
            MPSFIL5 = INFILE(1:1)//'.mps'
             OPEN(UNIT=10, FILE=MPSFIL5)
             status = impsmrd (MPSFIL5//char(0),
          .               pmac, pmar, pmat, pmae, penz,
          .               cstorsz, rstorsz, estorsz)
            SOLFILE5 = INFILE(1:1)//'.sol'
             OPEN(UNIT=20, FILE=SOLFILE5)
             status = slogfo (INFILE(1:1)//'.log')
            WRITE (20,*)  'This is the solution for  ',INFILE
          ELSEIF (INFILE(3:8) .EQ. ' ') THEN
            MPSFIL6 = INFILE(1:2)//'.mps'
             OPEN(UNIT=10, FILE=MPSFIL6)
             status = impsmrd (MPSFIL6//char(0),
          .               pmac, pmar, pmat, pmae, penz,
          .               cstorsz, rstorsz, estorsz)
            SOLFILE6 = INFILE(1:2)//'.sol'
             OPEN(UNIT=20, FILE=SOLFILE6)
             status = slogfo (INFILE(1:2)//'.log')
            WRITE (20,*)  'This is the solution for  ',INFILE
```

```
      ELSEIF (INFILE(4:8) .EQ. ' ') THEN
        MPSFIL7 = INFILE(1:3)//'.mps'
          OPEN(UNIT=10, FILE=MPSFIL7)
          status = impsmrd (MPSFIL7//char(0), pmac, pmar, pmat,
.  pmae, penz, cstorsz, rstorsz, estorsz)
        SOLFILE7 = INFILE(1:3)//'.sol'
          OPEN(UNIT=20, FILE=SOLFILE7)
          status = slogfo (INFILE(1:3)//'.log')
        WRITE (20,*)  'This is the solution for  ',INFILE
      ELSEIF (INFILE(5:8) .EQ. ' ') THEN
        MPSFIL8 = INFILE(1:4)//'.mps'
          OPEN(UNIT=10, FILE=MPSFIL8)
          status = impsmrd (MPSFIL8//char(0), pmac, pmar, pmat,
.  pmae, penz, cstorsz, rstorsz, estorsz)
        SOLFILE8 = INFILE(1:4)//'.sol'
          OPEN(UNIT=20, FILE=SOLFILE8)
          status = slogfo (INFILE(1:4)//'.log')
        WRITE (20,*)  'This is the solution for  ',INFILE
      ELSEIF (INFILE(6:8) .EQ. ' ') THEN
        MPSFIL9 = INFILE(1:5)//'.mps'
          OPEN(UNIT=10, FILE=MPSFIL9)
          status = impsmrd (MPSFIL9//char(0), pmac, pmar, pmat,
.  pmae, penz, cstorsz, rstorsz, estorsz)
        SOLFILE9 = INFILE(1:5)//'.sol'
          OPEN(UNIT=20, FILE=SOLFILE9)
          status = slogfo (INFILE(1:5)//'.log')
        WRITE (20,*)  'This is the solution for  ',INFILE
      ELSEIF (INFILE(7:8) .EQ. ' ') THEN
        MPSFIL10 = INFILE(1:6)//'.mps'
          OPEN(UNIT=10, FILE=MPSFIL10)
          status = impsmrd (MPSFIL10//char(0), pmac, pmar, pmat,
.  pmae, penz, cstorsz, rstorsz, estorsz)
        SOLFILE10 = INFILE(1:6)//'.sol'
          OPEN(UNIT=20, FILE=SOLFILE10)
          status = slogfo (INFILE(1:6)//'.log')
        WRITE (20,*)  'This is the solution for  ',INFILE(1:6)
      ELSEIF (INFILE(8:8) .EQ. ' ') THEN
        MPSFIL11 = INFILE(1:7)//'.mps'
          OPEN(UNIT=10, FILE=MPSFIL11)
          status = impsmrd (MPSFIL11//char(0), pmac, pmar, pmat,
.  pmae, penz, cstorsz, rstorsz, estorsz)
        SOLFILE11 = INFILE(1:7)//'.sol'
          OPEN(UNIT=20, FILE=SOLFILE11)
          status = slogfo (INFILE(1:7)//'.log')
          WRITE (20,*)  'This is the solution for  ',INFILE(1:7)
      ELSE
        MPSFIL12 = INFILE//'.mps'
        OPEN(UNIT=10, FILE=MPSFIL12)
       status = impsmrd (MPSFIL12//char(0), pmac, pmar, pmat,
.  pmae, penz, cstorsz, rstorsz, estorsz)
        SOLFILE12 = INFILE//'.sol'
          OPEN(UNIT=20, FILE=SOLFILE12)
       status = slogfo (INFILE//'.log')
          WRITE (20,*)  'This is the solution for  ',INFILE(1:8)
      ENDIF
*
      WRITE (*,'(//A)') 'Did You Use Hub Preferences In This Data Set?'
      WRITE (*,'(A/)') 'y or n'
```

```
      WRITE (*,'(A)') '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
      WRITE (*,'(A)') '!!If You Are Not Sure, Answer Yes!!'
      WRITE (*,'(A)') '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
        READ (*,'(A)')  HUBPREF
*
*  Check status for every function; non-zero value means failure.
********************************************************************
*      GET NODE LIMIT FOR INITIAL RUN
      WRITE (*,'(//A)') 'Enter nodelimit for initial solution process'
      WRITE (*,'(A)') 'Value must be integer between 1 and 1000000!!!'
      WRITE (*,'(A)') 'Suggested nodelimit appr. 10000'
        READ (*,'(I)')  nodelim
*
      newnode = 0
100   nodelo = 0
      nodehi = 1000000
*
      if (newnode.EQ.0) then status = sndlim(nodelim,nodelo,nodehi)
      else
          status = sndlim(newnode,nodelo,nodehi)
      endif
*
      if (status .ne. 0) goto 99000
      pobjdf = .099
      diflo = 0
      difhi = 100
*
      IF (HUBPREF .EQ. 'N' .OR. HUBPREF .EQ. 'NO' .OR. HUBPREF .EQ.
     c    'n' .OR. HUBPREF .EQ. 'no' .OR. HUBPREF .EQ. 'No')
     c          pobjdf = .99
      write(*,*) pobjdf
      status = sobjdf(pobjdf,diflo, difhi)
********************************************************************
*      SETTING Advance = 1
       advance = 1
       status = sadvin (advance)
************************************************
*         Solving the problem
      status = imipopt()
      if (status .ne. 0) goto 99000
*
      gstatus = gstat()
      write (*, 93) gstatus
      if (gstatus .eq. 103 .or. gstatus .eq. 106) then
       write (*,*)  'CANNOT FIND A FEASIBLE SOLUTION'
       write (20,*)  'CANNOT FIND A FEASIBLE SOLUTION'
       endif
*
      write (*,*)  'GSTAT = ',gstatus
*
      numvar = gmac()
      begin = 0
      end = numvar-1
      namelen = 8
      cstorsz=numvar*namelen
************************************************
      if (gstatus .eq. 105) then
      write (*,*)  'An INTEGER solution was found in ',nodelim+newnode,'
```

```
      . nodes.'
      write (*,*)  'It is not proven optimal!!!!'
      status = gmobj (obj)
      write (*,97) obj
*
      write(20,*)'An INTEGER solution was found in',nodelim+newnode,'
      . nodes.'
      write (20,*)  'It is not proven optimal!!!!'
      status = gmobj (obj)
      write (20,97) obj
* Writing the solution to the .sol file & the screen
            numvar = gmac()
      status = gmx(mipx,begin,end)
      status = gcname (store, cstorsz, surplus,
            .    begin, end, namelen)
*
      do 190 j = 0,numvar-1,1
        if (mipx(j) .ge. 0.0000001) then
          write (*, 96) store(j), mipx(j)
          write (20, 96) store(j), mipx(j)
        endif
190   continue
*
      WRITE (*,'(//A/)')'Dedicated C-141 missions in solution are:'
      WRITE (20,'(//A/)')'Dedicated C-141 missions in solution are:'
      do 200 j = 0,numvar-1,1
        if (mipx(j) .ge. 0.0000001) then
          if (store(j)(1:4) .eq. 'AC03') then
            write (*,96) store(j), mipx(j)
            write (20,96) store(j), mipx(j)
          endif
        endif
200     continue
      endif
*
      WRITE (*,'(//)')
      if (gstatus .ne. 103 .and. gstatus .ne. 106
   c      .and. gstatus .ne. 101) then
        WRITE(*,'(A)')'Would you like to continue searching for the ',
      .      'optimal solution?'
        WRITE (*,'(/A)')  'Y or N, then RETURN'
          READ (*,'(A)')  CONTINUE
        IF (CONTINUE .EQ. 'Y' .OR. CONTINUE .EQ. 'YES' .OR. CONTINUE .EQ.
   c    'y' .OR. CONTINUE .EQ. 'yes' .OR. CONTINUE .EQ. 'Yes') THEN
        write (20,'(//2A/)')  'The Next Solution is:'
        WRITE (*,'(A)') 'Enter Nodelimit for continued search.'
        WRITE (*,'(A)') 'Value must be integer between 1 and 1000000!!!'
        WRITE (*,'(A)') 'Suggested nodelimit increase appr. 100000'
          READ (*,'(I)')  newnode
*
*     SETTING Advance = 1
      advance = 1
      status = sadvin (advance)
      goto 100
      endif
      endif
*************************************************
      if (gstatus .eq. 101) then
```

```fortran
          status = gmobj(mipobj)
          if (status .ne. 0) goto 99000

          write (*, 94) mipobj
          write (*,*) 'Number of Nodes Used = ',gndc()
          write (*,*) 'Number of Iterations Used = ',gitcm()
*
          write (20, 94) mipobj
          write (20,*) 'Number of Nodes Used = ',gndc()
          write (20,*) 'Number of Iterations Used = ',gitcm()
*
          status = gmx(mipx,begin,end)
*             write(*,*)   'gmx status =', status
          status = gcname(store, cstorsz, surplus, begin, end, namelen)
*
          do 210 j = 0,numvar-1,1
            if (mipx(j) .ge. 0.0000001) then
              write (*, 96) store(j), mipx(j)
              write (20, 96) store(j), mipx(j)
            endif
210       continue
*
          WRITE(*,'(//A/)')'Dedicated C-141 missions in solution are:'
          WRITE(20,'(//A/)')'Dedicated C-141 missions in solution are:'
          do 220 j = 0,numvar-1,1
            if (mipx(j) .ge. 0.0000001) then
              if (store(j)(1:4) .eq. 'AC03') then
                WRITE (*,96) store(j), mipx(j)
                WRITE (20,96) store(j), mipx(j)
              endif
            endif
220       continue
          endif
*
99000 continue
*
*     Free space allocated by CPLEX and by reader
          status = ifreep ()
          status = ifreems ()
          stop
*
   90 format (' isolut status = 'i2/)
   93 format (' IP Solution status = ', i3/)
   94 format (' IP Solution value  = ', f15.6,///)
   95 format (' Variable Number', i5, ': Value = ', f15.6)
   96 format (' Variable ', a, ': Value = ', f15.6)
   97 format (' This solution is = ',f15.6//)
*
          end
```

APPENDIX C: User's Guide for Using mpsblr.f and solver.f

This appendix contains instructions, examples and recommendations for using the executable programs constructed from mpsbldr.f and solver.f.

After compiling the FORTRAN programs shown in Appendices A and B, executable programs will be produced for building and solving the mathematical formulation of the problem examined in this research. For the discussion in this appendix, these executable programs will be referred to as **mpsbldr.exe** and **solver.exe**. Screen outputs will be shown in a distinct font (Courier New).

User inputs will be shown in the same font, but will be preceded by "**Prompt:**" representing the operating system prompt. Specific commands will be surrounded by < and >. For example, the command <Return> means to press the Enter or Return key to enter the data previously typed. The questions regarding input values generally only require yes or no answers, followed by prompts to enter the specific values, if required. When answering "yes" or "no", you should enter either "Y" or "y", for "yes", and "N" or "n", for "no". Answering no will simply skip that section and proceed to the next question.

mpsbldr.exe

To begin the solution process, the user must build a Mathematical Programming System (MPS) file. This is accomplished by running mpsbldr.exe and providing the necessary inputs. To begin mpsbldr.exe simply type,

```
Prompt:  mpsbldr.exe <Return>
```

The screen will then show:

```
Enter the Filename for this data set.
!! Note:  Filename must be 8 letters or less!!
```

Input the filename you wish to call this data set:

```
Prompt:  example <Return>
```

The program will use the inputs that follow to build a file called "example.mps".

Additionally, a file named "example.dat" will be created that will contain the data that was

used to create the MPS file for future reference. You are then asked if you wish to change

the default number of airports or patient injury classes.

```
The Number of AOR Airports is currently = 4
The Number of HUB Airports is currently = 9
The Number of Patient Classes is currently = 6

Would you like to change any of these values?
Y or N, then RETURN
```

To change any of the values type:

```
Prompt:  Y <Return>
```

You will then be prompted to enter the appropriate number for the three areas.

For example, to change the number of AOR airports available to 2:

```
Please enter the number of AOR airports
```

```
Prompt: 2 <Return>
```

In the next steps, mpsbldr.exe will ask if AOR (and then HUB) airport names will

be entered. If desired, answer yes, and enter the appropriate names. The following

example shows what the screen will show and the inputs for entering Rhein-Main as the

name of AOR #1. For HUB names, the questions simply replace AOR with HUB.

```
Would you like to enter AOR names?
Y or N, then RETURN
```

```
Prompt:   Y <Return>

AOR #   1s names is:

Prompt:   Rhein-Main <Return>
```

Mpsbldr.exe will then continue through all AORs (or hubs) until all have been named. It will then ask if you desire to enter HUB preferences. HUB airports may be selected only as either preferred or not preferred. No other priority assignments are allowed. The following example shows how to select HUB #1 as a preferred HUB airport. The program will allow you to set preferences for all HUB airports.

```
Would you like to enter HUB preferences?
Y or N, then RETURN

Prompt:   Y <Return>

Do you PREFER to use HUB # 1?    Y or N

Prompt:   Y <Return>
```

Mpsbldr.exe, then asks you to enter any additional injury class names. The default injury classes are (in order): General Medical, Psychiatric, Surgical, Orthopedic, Spinal, and Burn.

```
PLEASE ENTER NEXT INJURY CLASS NAME.

Prompt:   NEUROSURGICAL <Return>
```

The patient injury classes will then be listed. Next you will be asked if you desire to use the default number of patients (and then available beds). The default values for the patients (and beds) can be found in the FORTRAN file mpsbldr.f which is printed in Appendix A. Following is an example of entering 100 as the number of General Medical patients at AOR #1. The program will continue through all injury classes and all AOR

(HUB) airports when entering patients (beds) until values for all AOR airport-injury class

(HUB airport-injury class) combinations have been entered.

```
Do you want to use test PATIENT values?
```

Prompt: N <Return>

```
***************************************************
Please enter the number of Patients at each AOR airport, per
    patient class.
***************************************************

***************************************************
AOR AIRPORT NUMBER   1

The Number of GEN. MEDICAL   Patients at AOR # 1 =
```

Prompt: 100 <Return>

The next question asks if you desire to change the flow-through capacity for the

HUB airports from the default value of 250 patients at each HUB airport. Note: different

flow-through capacities can be specified for individual HUB airports. The following

example shows how the flow-through capacity of HUB #1 is raised to 500. The program

will run through all available HUB airports.

```
The MAX number of Patients for Each HUB is 250 Patients.
Would You Like to Change This Value?
```

Prompt: Y <Return>

```
Please enter the Max number of patients allowed to transit
    each hub Airport.

The MAX Number of Patients Allowed to Transit HUB #  1 =
```

Prompt: 500 <Return>

After the flow-through capacities have been entered, mpsbldr.exe asks for the

number of available aircraft to be entered. The following example shows how to enter 5

as the number of available CRAF aircraft. Retrograde and Dedicated C-141s are entered

similarly.

```
There are currently     4  CRAF Aircraft,
```

```
                    10  Retrograde C-141s,
                     6 Dedicated C-141s.
```

```
Would you like to change the number of aircraft?
Y or N, then RETURN
```

Prompt: Y <Return>

```
Please enter the Number of CRAF Aircraft.
```

Prompt: 10 <Return>

Note: when no Dedicated C-141s are available, the solution process will then

search for optimum solutions which minimize the number of Retrograde C-141s required.

The last input values required to build the MPS file are the capacities of the

various aircraft. The following example shows how to update the capacity of the CRAF

aircraft from the default value of 100 to 80 patients.

```
The capacity of CRAF Aircraft is currently = 100
The capacity of Retrograde C-141s is currently = 60
The capacity of Dedicated C-141s is currently = 90
```

```
Would you like to change any of these values?
Y or N, then RETURN
```

Prompt: Y <Return>

```
Please enter the CRAF Aircraft Capacity
```

Prompt: 80 <Return>

Upon completion of entering the aircraft capacities, mpsbldr.exe outputs the MPS

file to "example.mps" and the corresponding input data to "example.dat". The solution

process may then be continued by solving the formulated problem using solver.exe.

solver.exe

After solver.f has been properly compiled, solver.exe is started by entering (from

the operating system prompt:

Prompt: solver.exe <Return>

The program begins by asking for the filename of the MPS file that contains the

formulation to be solved.

```
Enter the Filename to be solved.
!! Note:  Filename must be 8 letters or less!!
```

Prompt: example <Return>

Solver.exe continues by asking if HUB preferences were used in building the MPS

file.

```
Did You Use HuB Preferences In This Data Set?
y or n

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!IF You Are Not Sure, Answer Yes!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Because solver.exe uses different parameters in the solution process, it is important

that a yes answer is given to this question, if HUB preference were used in building the

MPS file. The program will not provide correct answers if HUB preferences were used,

but the above question is given a no answer. If HUB preferences were not used, and a yes

answer is given, the answer will be correct but will probably take longer to converge to

the *best* solution. Note: if no hub preferences are used, all hubs are considered *preferred*.

Prompt: Y <Return>

Next you are instructed to enter the initial node limit to be used by solver.exe in

the solution process. As shown on the screen, the suggested node limit is 10000 nodes.

Using this node limit, the solver.exe will stop upon reaching an optimal integer solution or

after evaluating 10000 nodes. Note: 10000 nodes takes approximately 8.5 minutes.

Since the first *good* integer answer is generally not found in less than around 1000 nodes,

an initial node limit greater than 1000 is highly recommended.

```
Enter the nodelimit for the initial solution process.
```

```
Value must be integer between 1 and 1000000!!!
Suggested nodelimit appr. 10000
```

Prompt: 10000 <Return>

The screen will then show solution information, including node number, the

objective function value at that node, and the current *best* integer objective function value.

An "*" preceding a node number indicates that an integer solution has been found at that

node.

Upon reaching an optimal integer solution, solver.exe stops and prints to the

screen and a solution file ("example.sol") the variable names and the values for the non-

zero variables. In addition, the missions that are required to be flown by dedicated C-141

aircraft are distinctly listed. An example of a solution file is shown in Appendix E.

If an optimal integer solution has not been found by the node limit initially set,

solver.exe outputs the *best* integer solution found, as described above for the optimal

solution case. It also then asks if the search is to be continued. If so, the additional

number of nodes that will be used in this continued search must be entered. Again a

suggested node limit is shown. If time is available (approximately 1.5 hours), it is

recommended to use a node limit of 100000. While an optimal solution is not guaranteed

to be found in less than 100000 nodes, it will provide a very good forecast/schedule near

the optimal solution. The following example shows how the continued node limit is

entered.

```
Would you like to continue searching for the
optimal solution?

Y or N, then RETURN
```

Prompt: Y <Return>

```
Enter Nodelimit for continued search.
```

```
Value must be integer between 1 and 1000000!!!
Suggested nodelimit appr. 100000
```

Prompt: 100000 <Return>

Solver.exe uses this updated node limit to continue the search for an optimal

solution. Upon reaching an optimal integer solution, or the specified node limit, the

output and required user inputs are identical to those described for the initial nodelimit.

Recommendations

Start with no HUB preferences. There are many different possible aircraft type-

route combinations that will use the same number of dedicated C-141s. When no HUB

preferences are specified, solver.exe will simply find one that minimizes the number of

dedicated C-141s required. When HUB preferences are specified, it is forced to find one

that also minimizes the total number of all aircraft flown to non-preferred HUB airports.

Therefore, the solution process finds an optimal forecast/schedule, with the minimum

number of dedicated C-141s required, much more quickly when HUB preferences are not

specified. After finding this optimal forecast, the user should then specify any necessary

HUB preferences. Then the solutions given by solver.exe may be examined to ensure that

the given forecast is using the fewest number of dedicated C-141s. Using whatever time is

available, the user may then have solver.exe search for the aircraft type-route

combinations to reduce the number of aircraft being flown to non-preferred HUB airports.

In general, when HUB preferences are used, the tens and digits places will show the

number of Dedicated C-141s required, and the one-tenths place will show the total

number of aircraft flown into non-preferred HUB airports. For completeness, the solution

file should be examined.

Additionally, using only the required aircraft shown in the initial run (with no HUB

preferences specified) may speed up the solution process.

APPENDIX D: Test Data

This appendix contains a compilation of the test data sets used in the analysis

presented in Chapter 4.   The data concerning patients and test beds for this research was

obtained from Major Mike Loftus' thesis (10).

- Only 6 patient injury classes were used in each test case.

- Aircraft capacities used in all test cases are:
      CRAF Aircraft   - 100 patients
      Retrograde C-141s -   60 patients
      Dedicated C-141s   - 90 patients

- All 9 hubs are used in each test case.

Patient Totals

AOR	PATIENT CLASS					
	GEN. MEDICAL	PSYCHIATRIC	SURGICAL	ORTHOPEDIC	SPINAL	BURN
1	43	10	143	120	10	3
2	43	10	143	120	5	5
3	46	11	161	132	11	4
4	37	5	167	120	13	7

Bed Totals

HUB	PATIENT CLASS					
	GEN. MEDICAL	PSYCHIATRIC	SURGICAL	ORTHOPEDIC	SPINAL	BURN
1	5244	1519	3881	1184	597	486
2	1583	665	2258	703	213	145
3	4006	413	2925	444	102	61
4	3054	1277	2690	1177	239	202
5	1547	434	1626	936	73	29
6	9412	2428	7495	2281	479	236
7	5903	1482	4017	1277	146	88
8	3988	1084	3311	1228	189	169
9	5811	1729	6957	1946	360	290

Test Data (cont.)

Test Case Number	# of AORs Used	# of Available Aircraft CRAF	C-141R	C-141D	Pref HUB 1	2	3	4	5	6	7	8	9	HUB Flow 1	2	3	4	5	6	7	8	9
1	4	4	10	6										250 for all hubs								
2	4	4	10	6						X	X	X	X	250 for all hubs								
3	4	4	10	6	X	X	X	X						250 for all hubs								
4	4	4	10	6	X		X		X		X		X	250 for all hubs								
5	4	4	14	3										250 for all hubs								
6	4	6	16	0										250 for all hubs								
7	3	4	10	6										250 for all hubs								
8	2	4	10	6										250 for all hubs								
9	1	4	10	6										250 for all hubs								
10	4	4	10	6										300 for all hubs								
11	4	4	10	6										400 for all hubs								
12	4	4	10	6										500 for all hubs								
13	4	4	10	6										1000 for all hubs								
14	4	4	10	6				X						250	500	250	500	250	500	250	500	250
15	4	4	10	6		X				X		X		250	500	250	500	250	500	250	500	250
16	4	4	10	6						X	X	X	X	500 for all hubs								

Preferred HUBS (X = Not Preferred)

APPENDIX E: Solution File Examples

This appendix contains three examples of the solution files output by solver.f: test cases 1, 2 and 9. Note test case 1 returns only one set of variables (for the optimal solution). Test case 2 has two sets of variables. The first set is for the best solution found within the initial 10000 node limit. The second is for the best solution found within 30000 nodes. Since neither of these solutions for test case 2 are optimal, a warning is supplied prior to the variable values to show the solution is not proven optimal. Analysis of test case 9 shows that while only nine of the 10 available Retrograde C-141s were used, five Dedicated C-141s would still be required.

The indices for the variables, are described in the following table.

.Aircraft Variables			
AC	01	01	01
	A/C Type #1	From AOR #1	To HUB #1
Patient Variables			
P	1	01	01	01
	At AOR #1	Injury Class #1	Flown On A/C Type #1	To Hub #1
Non-Evacuated Patients			
NOEV	01	01	
	At AOR #1	Injury Class #1	

test1.sol
```
This is the solution for  test1
 IP Solution value  = 5.0000

Variable AC010104: Value =        1.000000
Variable AC020103: Value =        1.000000
Variable AC030101: Value =        1.000000
Variable AC030109: Value =        1.000000
Variable AC010204: Value =        1.000000
Variable AC020202: Value =        3.000000
Variable AC030203: Value =        1.000000
Variable AC010302: Value =        1.000000
Variable AC010309: Value =        1.000000
Variable AC020301: Value =        3.000000
```

```
Variable AC020404: Value =          1.000000
Variable AC020408: Value =          2.000000
Variable AC030403: Value =          1.000000
Variable AC030406: Value =          1.000000
Variable P0110104: Value =         43.000000
Variable P0120104: Value =         10.000000
Variable P0130104: Value =         32.000000
Variable P0130203: Value =         27.000000
Variable P0130301: Value =         84.000000
Variable P0140203: Value =         30.000000
Variable P0140309: Value =         90.000000
Variable P0150104: Value =         10.000000
Variable P0160203: Value =          3.000000
Variable P0210104: Value =         13.000000
Variable P0210202: Value =         30.000000
Variable P0220303: Value =         10.000000
Variable P0230104: Value =         77.000000
Variable P0230303: Value =         66.000000
Variable P0240202: Value =        120.000000
Variable P0250104: Value =          5.000000
Variable P0260104: Value =          5.000000
Variable P0310102: Value =         16.000000
Variable P0310201: Value =         30.000000
Variable P0320102: Value =         11.000000
Variable P0330102: Value =         73.000000
Variable P0330109: Value =         88.000000
Variable P0340201: Value =        132.000000
Variable P0350109: Value =         11.000000
Variable P0360201: Value =          4.000000
Variable P0410204: Value =         37.000000
Variable P0420204: Value =          5.000000
Variable P0430303: Value =         77.000000
Variable P0430306: Value =         90.000000
Variable P0440208: Value =        120.000000
Variable P0450204: Value =         13.000000
Variable P0460303: Value =          7.000000
```

The Dedicated C-141 missions in this solution are:

```
Variable AC030101: Value =          1.000000
Variable AC030109: Value =          1.000000
Variable AC030203: Value =          1.000000
Variable AC030403: Value =          1.000000
Variable AC030406: Value =          1.000000
```

---

test2.sol
```
This is the solution for   test2
An INTEGER solution was found in  10000 nodes.
It is not proven optimal!!!!
 This solution is =         6.200000


Variable AC020102: Value =          1.000000
Variable AC020103: Value =          1.000000
Variable AC020104: Value =          1.000000
```

```
Variable AC020105: Value =         2.000000
Variable AC030105: Value =         1.000000
Variable AC010201: Value =         1.000000
Variable AC010202: Value =         1.000000
Variable AC010208: Value =         1.000000
Variable AC030203: Value =         1.000000
Variable AC020303: Value =         1.000000
Variable AC020306: Value =         1.000000
Variable AC030303: Value =         2.000000
Variable AC030304: Value =         1.000000
Variable AC010404: Value =         1.000000
Variable AC020401: Value =         2.000000
Variable AC020405: Value =         1.000000
Variable AC030402: Value =         1.000000
Variable P0110204: Value =        43.000000
Variable P0120204: Value =        10.000000
Variable P0130202: Value =        57.000000
Variable P0130305: Value =        86.000000
Variable P0140205: Value =       120.000000
Variable P0150203: Value =         3.000000
Variable P0150204: Value =         7.000000
Variable P0160202: Value =         3.000000
Variable P0210101: Value =        11.000000
Variable P0210303: Value =        32.000000
Variable P0220101: Value =        10.000000
Variable P0230101: Value =        43.000000
Variable P0230108: Value =       100.000000
Variable P0240101: Value =        20.000000
Variable P0240102: Value =       100.000000
Variable P0250101: Value =         5.000000
Variable P0260101: Value =         5.000000
Variable P0310203: Value =        39.000000
Variable P0310206: Value =         7.000000
Variable P0320206: Value =        11.000000
Variable P0330303: Value =       161.000000
Variable P0340206: Value =        42.000000
Variable P0340304: Value =        90.000000
Variable P0350303: Value =        11.000000
Variable P0360203: Value =         4.000000
Variable P0410104: Value =         6.000000
Variable P0410205: Value =        31.000000
Variable P0420104: Value =         5.000000
Variable P0430104: Value =        77.000000
Variable P0430302: Value =        90.000000
Variable P0440104: Value =         5.000000
Variable P0440201: Value =       115.000000
Variable P0450205: Value =        13.000000
Variable P0460104: Value =         7.000000


The Dedicated C-141 missions in this solution are:
Variable AC030105: Value =         1.000000
Variable AC030203: Value =         1.000000
Variable AC030303: Value =         2.000000
Variable AC030304: Value =         1.000000
Variable AC030402: Value =         1.000000


The Next Solution is:
```

```
An INTEGER solution was found in  40000 nodes.
It is not proven optimal!!!!
  This solution is =          5.200000

Variable AC020104: Value =          2.000000
Variable AC020105: Value =          2.000000
Variable AC030105: Value =          1.000000
Variable AC010201: Value =          1.000000
Variable AC010204: Value =          1.000000
Variable AC010208: Value =          1.000000
Variable AC030205: Value =          1.000000
Variable AC010301: Value =          1.000000
Variable AC030303: Value =          2.000000
Variable AC030306: Value =          1.000000
Variable AC020401: Value =          1.000000
Variable AC020402: Value =          4.000000
Variable AC020403: Value =          1.000000
Variable P0110204: Value =         43.000000
Variable P0120204: Value =         10.000000
Variable P0130204: Value =         53.000000
Variable P0130305: Value =         90.000000
Variable P0140205: Value =        120.000000
Variable P0150204: Value =         10.000000
Variable P0160204: Value =          3.000000
Variable P0210101: Value =          3.000000
Variable P0210305: Value =         40.000000
Variable P0220101: Value =         10.000000
Variable P0230104: Value =         43.000000
Variable P0230108: Value =        100.000000
Variable P0240101: Value =         67.000000
Variable P0240104: Value =         53.000000
Variable P0250101: Value =          5.000000
Variable P0260101: Value =          5.000000
Variable P0310306: Value =         46.000000
Variable P0320306: Value =         11.000000
Variable P0330303: Value =        128.000000
Variable P0330306: Value =         33.000000
Variable P0340101: Value =         85.000000
Variable P0340303: Value =         47.000000
Variable P0350101: Value =         11.000000
Variable P0360101: Value =          4.000000
Variable P0410201: Value =         37.000000
Variable P0420201: Value =          5.000000
Variable P0430201: Value =         18.000000
Variable P0430202: Value =        149.000000
Variable P0440202: Value =         60.000000
Variable P0440203: Value =         60.000000
Variable P0450202: Value =         13.000000
Variable P0460202: Value =          7.000000

The Dedicated C-141 missions in this solution are:
Variable AC030105: Value =          1.000000
Variable AC030205: Value =          1.000000
Variable AC030303: Value =          2.000000
Variable AC030306: Value =          1.000000
```

test9.sol

This is the solution for test9
IP Solution value =   5.000000

Variable AC010102: Value =   1.000000
Variable AC010103: Value =   1.000000
Variable AC010106: Value =   1.000000
Variable AC010109: Value =   1.000000
Variable AC020102: Value =   1.000000
Variable AC020103: Value =   1.000000
Variable AC020105: Value =   4.000000
Variable AC020106: Value =   1.000000
Variable AC020107: Value =   1.000000
Variable AC020108: Value =   1.000000
Variable AC030103: Value =   1.000000
Variable AC030104: Value =   3.000000
Variable AC030106: Value =   1.000000
Variable P0110106: Value =   39.000000
Variable P0110207: Value =   40.000000
Variable P0110306: Value =   90.000000
Variable P0120303: Value =   36.000000
Variable P0130106: Value =   22.000000
Variable P0130109: Value =   58.000000
Variable P0130202: Value =   60.000000
Variable P0130203: Value =   60.000000
Variable P0130205: Value =   240.000000
Variable P0130206: Value =   60.000000
Variable P0130208: Value =   60.000000
Variable P0130303: Value =   54.000000
Variable P0140102: Value =   100.000000
Variable P0140103: Value =   100.000000
Variable P0140109: Value =   42.000000
Variable P0140304: Value =   250.000000
Variable P0150106: Value =   39.000000
Variable P0160207: Value =   19.000000

The Dedicated C-141 missions in this solution are:
Variable AC030103: Value =   1.000000
Variable AC030104: Value =   3.000000
Variable AC030106: Value =   1.000000

Bibliography

1. Alfano, Joseph P. and John C. O'Neill. *Wartime CONUS Casualty Distribution System Using Dedicated CRAF Aircraft*, MS Thesis/AFIT/GST/OS/85M, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, March 1985 (AD-A156076).

2. Carnegie Group, Inc. *TRAC2ES CONOPS*, prepared for USTRANSCOM GPMRC, Pittsburgh, PA, 26 October 1994.

3. CPLEX 3.0. *Using the CPLEX Callable Library*, Incline Village, NV: CPLEX Optimization, Inc., 1994.

4. Department of Defense. *Armed Services Medical Regulating*, DOD Directive 5154.6, Washington: GPO, 29 April 1993.

5. Department of the Air Force. *Annex Q to AMC Omnibus OPLAN*, Headquarters Air Mobility Command, Scott AFB, IL, 1 June 1994.

6. Department of the Air Force. *MAC Aeromedical Evacuation (AE) Concept of Operations Phase II*, Prepared by Analytical Systems Engineering Corporation. Contract F19628-86-D-003, 15 June 1989.

7. Dobbs, Rebecca and others. *Concept Development for an Aeromedical Data Acquisition and Communication System (AMDACS)*, Interim Report, USAFSAM-TR-90-20, Battelle Columbus Division, Columbus OH, August 1990.

8. Durham, Randy P. *Red Cross Rising*, Airlift: 5-9 (Summer 1988).

9. Klotz, Ed. Technical Support Representative, CPLEX Optimization Inc., personal correspondence, January 1995.

10. Loftus, Michael J. *Patient Scheduling and Aircraft Routing for Strategic Aeromedical Evacuation*, MS Thesis/AFIT/GST/ENS/93M-7, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, March 1985 (AAJ-7546).

11. Mahlum, Phil. GPMRC, USTRANSCOM, telephone interviews, January 1995.

12. -----. Electronic facsimile, 30 January 1995.

13. Nemhauser, George L. And Laurence A. Wolsey. *Integer and Combinatorial Optimization*, New York: John Wiley and Sons, 1988.

14. Parker, R. Gary and Ronald L. Rardin. *An Overview of Complexity Theory in Discrete Optimization*, Part 1, IIE Transactions, March 1982.

15. -----. *An Overview of Complexity Theory in Discrete Optimization*, Part 2, IIE Transactions, June 1982

16. Secretary of Defense. *Policy with Respect to Evacuation of Patients*, 7 September 1949, memorandum as quoted by Brig Gen Dan Ogle, special assistant to the surgeon general, to the commanding general, Air University, 4 November 1949.

17. Schwartz, Rudy. *Aeromedical Evacuation Contingency Planning*, Research Paper, Naval War College, June 1992.

18. Wolfe, Charles R. *The Use of Simulation to Evaluate Strategic Aeromedical Evacuation Policy and Planning*, MS Thesis/AFIT/GOR/ENS/93M-26, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, March 1993.

Vita

Captain Donald F. Kimminau was born on 13 September 1962 in Omaha,

Nebraska. He graduated from Lewis-Palmer High School in Monument, Colorado in

1980 and attended the University of Colorado in Boulder, Colorado, graduating with a

Bachelor's of Science in Applied Mathematics. He received his United States Air Force

commission through the Reserve Officer Training Corps in December 1984. After

graduating from Undergraduate Pilot Training at Reese AFB, Texas, in 1986, he served

as First Pilot in the C-12 aircraft at Osan AB, Republic of South Korea. From Osan AB,

he was assigned to Nellis AFB, Nevada from 1987 to 1990. At Nellis, he flew as a C-12

Instructor Aircraft Commander and was the Chief of Scheduling and Chief of Training.

His next assignment was to Rhein-Main AB, Germany, from 1990 to 1993, where he was

a C-130 Aircraft Commander and served in Desert Shield/Desert Storm, Provide Comfort,

and Provide Hope among other duties. He was also the 435th Airlift Wing's Nuclear

Surety Officer, and a C-130 Prime Nuclear Airlift Aircraft Commander. He entered the

Air Force Institute of Technology's Graduate School of Engineering, in August 1993.

Permanent Address: 107 East Horizon Circle
Tucson, AZ 85737